MICROCOMPUTADOR CURSO PRÁTICO

TIRE O MÁXIMO DO MICRO EM 30 SEMANAS

PUBLICAÇÃO SEMANAL ILUSTRADA

14

Cr$ 8.200

Printer/plotters • Controle remoto
Técnicas de otimização

rioGráfica

MANAUS, ALTAMIRA, MACAPÁ, JI.PARANÁ, PORTO VELHO, RIO BRANCO, SANTARÉM, BOA VISTA, SINOP, ALTA FLORESTA: Cr$ 10.660

MICROCOMPUTADOR CURSO PRÁTICO

APLICAÇÕES

Controle por computador 277

PROJETOS DE PROGRAMAS

Travessia do Deserto 280

PERIFÉRICOS

Printer/plotters 282

RAIO X

BBC modelo B 285

LINGUAGENS

Dimensões matriciais 288

SOFTWARE

Aplicativos 291

CIÊNCIA DA COMPUTAÇÃO

No nível 292

TÉCNICAS DE PROGRAMAÇÃO

Métodos de otimização 294

PERFIL

Lógica empresarial 296

TESTE

Depois de ler com atenção todo o fascículo, responda a estas perguntas. Veja a solução no próximo número, junto com um novo teste.

1) Qual o nome do dispositivo essencial para que um micro controle aparelhos eletromecânicos? Trata-se de uma chave que, embora acionada por pulsos de baixa intensidade, pode ligar e desligar correntes elétricas mais altas.
2) De que maneira um printer/plotter imprime caracteres de texto?
3) Por que o uso de variáveis inteiras, e não reais, acelera a execução de um programa?
4) Como se chama o bit que se acrescenta a um código binário para assegurar a correção da transmissão dos dados?

RESPOSTAS DO TESTE ANTERIOR

1) As pesquisas de inteligência artificial organizam o conhecimento humano a respeito de certas áreas em sistemas denominados "especialistas".
2) O Compaq Plus é relativamente lento no acesso a dados porque seu processador 8088, embora opere em 16 bits, possui um bus de dados de apenas 8 bits.
3) O comando CMD no Commodore serve para direcionar o conteúdo da tela para outro canal de saída, como a impressora.
4) O elemento principal de um traçador digital é o potenciômetro.

PLANO DA OBRA

MICROCOMPUTADOR - CURSO PRÁTICO é uma coleção de 30 fascículos de periodicidade semanal. Os 29 primeiros terão, cada um, vinte páginas internas (miolo) e quatro capas, além de uma introdução de dezesseis páginas acompanhando o fascículo 1. O fascículo 30 incluirá o índice geral da obra. Os miolos, encadernados, constituem dois volumes de um curso prático de microcomputação, com seções indicadas por tarjas de diferentes cores.
As duas primeiras capas são descartáveis; as 3.ª e 4.ª trazem programas de jogos. Encadernadas, formam um volume de cem páginas.

COMO ENCADERNAR

As capas duras, incluindo as guardas, estarão à venda simultaneamente com os fascículos 10 (**Volume 1**) e 30 (**Volume 2** e **Jogos**).
Volume 1 - Deve ser encadernado com os elementos dispostos nesta ordem: guardas; frontispício e introdução (publicados com o fascículo 1); miolos dos fascículos de 1 a 15; e guardas.
Volume 2 - Guardas; frontispício (que virá com o fascículo 30); miolos dos fascículos de 16 a 30; e guardas.
Jogos - Guardas; frontispício e sumário (que constituem as 3.ª e 4.ª capas do fascículo 30); 3.ª e 4.ª capas dos fascículos de 1 a 29; e guardas.

rioGráfica **Diretoria Executiva:** Oscar Neves; Nilo Sérgio de Almeida; Danilo Esteves Costa; **GRUPO EDUCAÇÃO E CULTURA - Planejamento e Comercialização - Gerente de Planejamento:** João Azeredo Coutinho; **Gerente de Marketing:** Jaime Rodrigues; **Gerente de Produto:** José Mauro Porto; **Gerente do P.C.P.:** Aylton Menezes; **Diretor de Vendas:** Marcos March; **Gerente de Vendas:** Rubens Barbosa; **Coordenadores de Promoção:** Paulo Cesar M. Seixas; Edgar Mello M. Neto; **Gerente de Circulação:** Norberto Martin; **Supervisor de Operações:** Abel Pereira Pinto; **Editorial - Diretor:** João J. Noro; **Editor-chefe:** Mauricio Rittner; **Administração Editorial:** Heitor de Souza Paixão; **Editor:** Jesse Navarro; **Revisores:** Isis Augusta Loyolla; Cacilda Guerra; Luiz Vicente Vieira Filho; Marcos Domingos Agathão; Maria da Graça Mendonça Couto; **Diretor de Arte:** Adalberto Cornavaca; **Chefe de Arte:** Bonifácio Duardes Miranda; **Chefe de Estúdio:** José Yuji Kuribayashi; **Assistente de Arte:** Miguel Luis Escámez Simón; **Colaboradores - Editores-assistentes:** Claudio Marcondes; Isa Mara Lando; Luiz Carlos Pizarro Marin; **Preparação de Texto:** Luiz Carlos Cardoso; **Tradução:** Fernando Dortman Knijnik; Lúcia Maria Leal Gonçalves; Maria Clara Cescato; Maria Ester Menezes Martinho; Newton Roberval Eichemberg; Patricia de Paiva e Castro; Regina Amarante; **Pesquisa:** Stella Maria Quentel; **Arte:** Edemilson Ribeiro da Silva; **Assessoria Técnica:** Geraldo Coen; Gustavo José Ferreira Grubba; Lauro de Lauro; Pierluigi Piazzi; Renato F. Cosentino; Sérgio Jorge Batarce; Imarés Microcomputadores. As fotos não creditadas pertencem à obra original. Seguem-se os créditos das demais: Fernando Scavone.

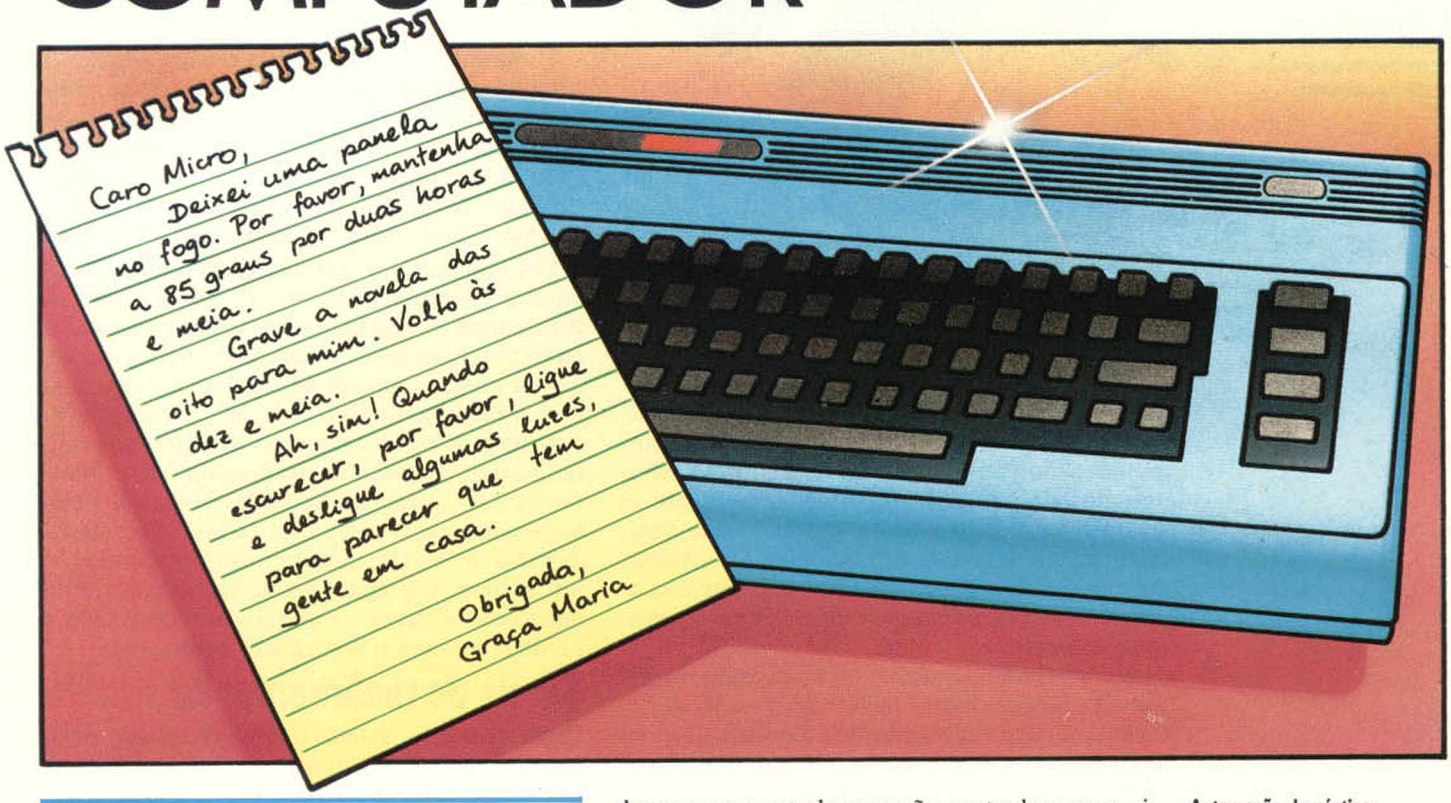
© APSIF Copenhagen, 1984; © Orbis Publishing Ltd., 1984; © Editora Rio Gráfica Ltda., 1985, para a língua portuguesa.

Composição: Editora Rio Gráfica Ltda. e AM Produções Gráficas Ltda. **Impressão:** JBIG.

NÚMEROS ATRASADOS - Você poderá comprar os exemplares que faltam em sua coleção, pelo preço do último fascículo posto à venda, em sua banca de jornal preferida. Caso não consiga obtê-los, faça sua solicitação por carta endereçada diretamente à Editora Rio Gráfica Ltda., rua Itapiru, 1209, Rio Comprido, Rio de Janeiro, CEP 20251. O atendimento será feito por via postal. Nas cidades do Rio de Janeiro e de São Paulo, as compras poderão ser efetuadas pessoalmente nos seguintes endereços: Rio de Janeiro - Rua Itapiru, 1209, Rio Comprido; São Paulo - Rua Frei Caneca, 1152, Consolação.

Distribuidor exclusivo para todo o Brasil: Fernando Chinaglia - Rua Teodoro da Silva, 907, Rio de Janeiro. Distribuidor para Portugal: Electroliber Lda. - Rua Prof. Reinaldo dos Santos, 1488, Lisboa.

CONTROLE POR COMPUTADOR



Os usuários de micros têm a opção de desenvolver seu próprio software, mas poucos pensariam em construir periféricos em casa. Agora já existem componentes que tornam isso possível em alguns casos.

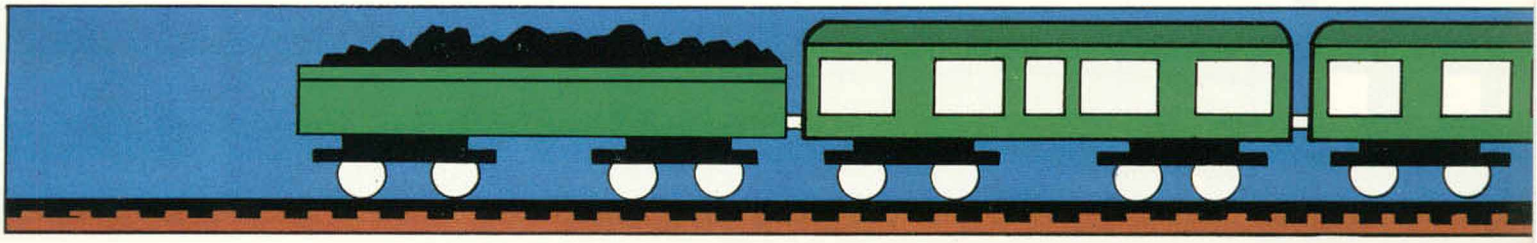
O único motivo pelo qual as pessoas programam um computador para abrir as cortinas da casa pela manhã ou regar as plantas quando saem de férias é que isso as diverte. E nada há de errado em fazer uma coisa pela simples razão de ser divertido. Quem escreve seus próprios programas por hobby não gasta horas e horas nisso apenas por prazer?

No momento, construir periféricos por conta própria não passa de uma brincadeira, mas em breve essa atividade poderá se revelar bastante lucrativa. Muita gente acredita que, num futuro próximo, robôs e outros dispositivos controlados por computadores serão parte de nossas vidas, da mesma forma como os micros tornaram-se lugar-comum a partir de 1980.

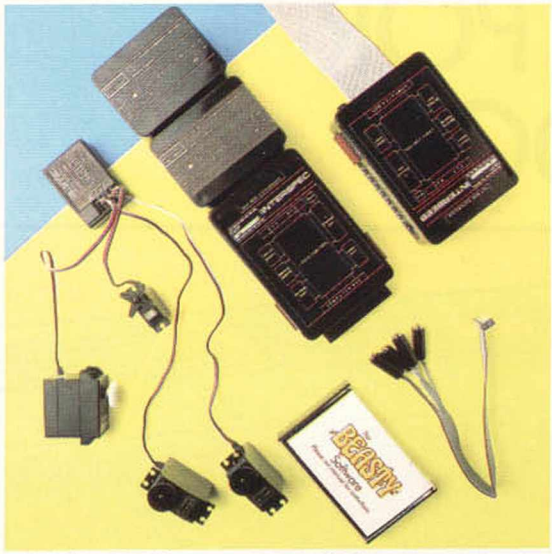
Portanto, a habilidade desenvolvida hoje poderá ter inestimável valor no futuro. Afinal, grandes indústrias de computadores, como a Apple e a Atari, começaram em garagens de pessoas que estavam apenas "brincando" com componentes e aparelhos eletrônicos.

Alguns elementos são indispensáveis em qualquer sistema controlado por computador. Além deste e do aparelho a ser controlado, precisa-se também de algum meio físico, para transmissão das mensagens de controle do micro ao aparelho, e de um software que decida quais serão essas mensagens. Isso é apenas o começo: o computador deverá ter, ainda, um meio de medir os efeitos de seu controle, para proceder aos ajustes necessários. Sem esse feedback (mecanismo de realimentação), o micro será tão inútil quanto um motorista de olhos vendados.

Automação doméstica
A conexão do micro a aparelhos domésticos permite a realização automática de inúmeras operações. Previamente programado, o computador reage a fatos como, por exemplo, queda de temperatura ou presença de ladrões ligando o aquecedor ou acionando o alarma.



Conexões de controle
Vários micros já dispõem de interfaces apropriadas para o controle de aparelhos. Em geral são dispositivos de chaveamento baseados em relés. O computador liga ou desliga a corrente de determinado equipamento e depois confirma, por um sensor, se ele está ou não funcionando.



Todos os sistemas controlados por computador funcionam com base em sinais elétricos. Mas os sinais (digitais) utilizados no interior do computador são fracos demais para uso direto. Mesmo uma simples lâmpada utiliza quantidade de energia consideravelmente maior do que qualquer sinal interno de um computador. Assim, precisamos de algum dispositivo que converta as "baixas voltagens" do computador em "voltagens maiores".

Num primeiro estágio, o computador necessita de canal de comunicação com o exterior. Para isso, alguns micros dispõem de saída — serial ou paralela — especial para o controle de dispositivos externos. Controladas pelo microprocessador, essas saídas (ou portas) possuem uma área reservada de endereçamento na memória. A leitura ou gravação, a partir do exterior, de algum dado nessa parte da memória não afeta o funcionamento do micro.

Sozinha, a saída para os dispositivos externos pode ser usada para acender e apagar LEDs (diodos emissores de luz), mas, para outras finalidades, são necessários mais componentes. Às vezes, convém acoplar alguns componentes eletrônicos e mais uma pequena fonte de energia elétrica para tornar possível o controle de relés. Um relé não passa de uma chave que liga e desliga correntes elétricas relativamente altas, mas pode ser controlado por um pulso elétrico de baixa intensidade. Constitui, portanto, uma das melhores maneiras de converter os fracos pulsos elétricos do computador em correntes utilizáveis.

Um sistema que permita utilizar relés está entre os primeiros objetivos de qualquer pessoa com intenção de controlar dispositivos eletromecânicos. Por meio de relés, acionam-se diferentes aparelhos: desde motores elétricos, bombas hidráulicas e campainhas, até trenzinhos e carrinhos elétricos radiocontrolados. A maior parte dos relés usados por entusiastas dos inventos caseiros só funciona com aparelhos movidos a pilha, mas isso é suficiente para a maioria dos projetos.

O chaveamento da corrente elétrica permite o controle, pelo computador, de aquecedores, lâmpadas potentes e inúmeros outros eletrodomésticos. Ele também possibilita ao computador atuar como timer, que liga a televisão no horário de um programa, ou acende e apaga as luzes da casa à noite para desencorajar os ladrões.

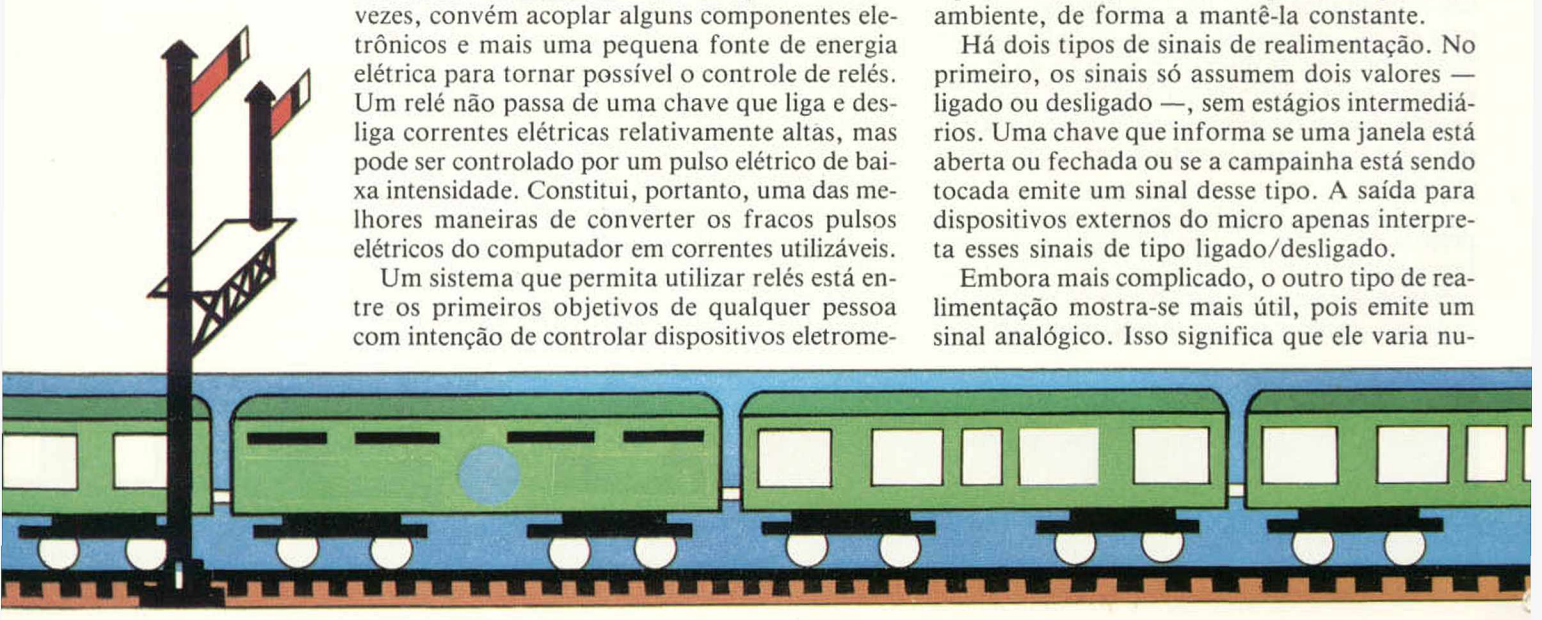
Por enquanto, o computador precisa ser conectado diretamente ao aparelho controlado. A fim de superar essa limitação, várias companhias estão desenvolvendo dispositivos para permitir que os fios condutores da corrente elétrica também transmitam as informações de controle. O sistema requer um computador central que emita sinais para unidades escravas em outros lugares da casa, ligadas a tomadas normais. Essas mensagens ligariam ou desligariam cada unidade escrava. Assim, qualquer eletrodoméstico — um abajur, um aparelho de televisão, ou um aquecedor elétrico — poderia ser conectado à unidade escrava e controlado pelo micro.

Sinais de realimentação

Quase todos os sistemas controlados por computador necessitam de algum tipo de feedback que avalie o funcionamento do conjunto. Aparentemente não há necessidade de realimentação quando o micro está apenas acendendo e apagando luzes. Mas seria muito melhor se ele pudesse detectar quando está escurecendo e ligar as lâmpadas da casa em resposta a essa informação. Também seria muito útil se ele percebesse a entrada de alguém num aposento e acendesse as luzes. Ou, ainda, quando está controlando um aquecedor, ele precisa saber qual a temperatura ambiente, de forma a mantê-la constante.

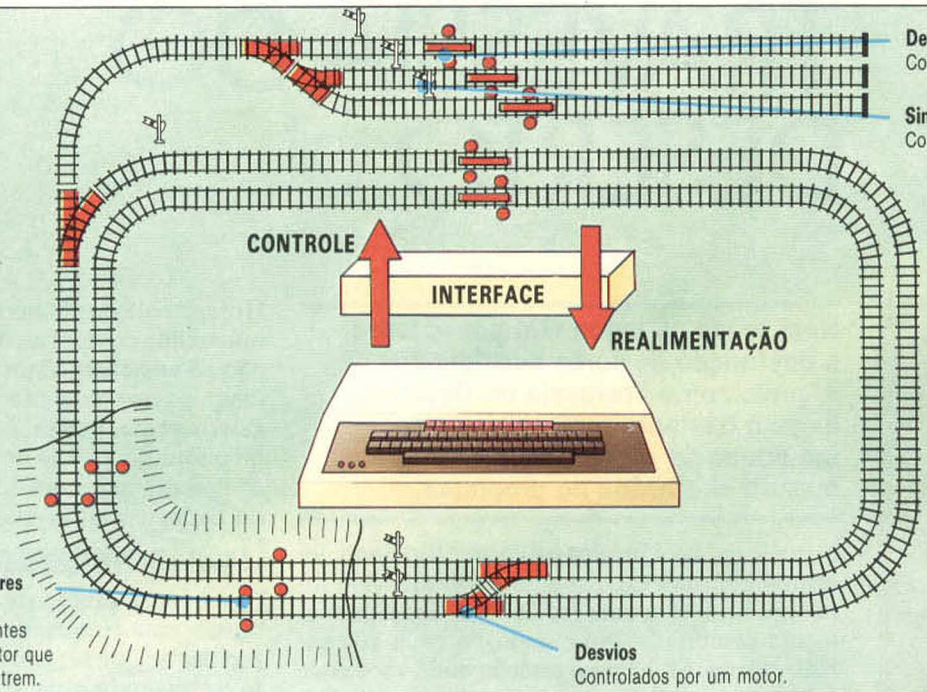
Há dois tipos de sinais de realimentação. No primeiro, os sinais só assumem dois valores — ligado ou desligado —, sem estágios intermediários. Uma chave que informa se uma janela está aberta ou fechada ou se a campainha está sendo tocada emite um sinal desse tipo. A saída para dispositivos externos do micro apenas interpreta esses sinais de tipo ligado/desligado.

Embora mais complicado, o outro tipo de realimentação mostra-se mais útil, pois emite um sinal analógico. Isso significa que ele varia nu-



O chefe da estação

No controle de qualquer equipamento por micro — de um trenzinho elétrico a uma casa inteira — usa-se a mesma técnica. Monta-se um loop, no qual o computador emite sinais de controle, por meio de uma interface, ao dispositivo conectado. Quando esses sinais, que controlam servomotores, luzes etc. retornam, as informações são captadas por sensores como células fotoelétricas ou chaves liga/desliga. Essa realimentação permite que o micro controle o aparelho com precisão.



Desacoplador do trem
Controlado por um motor.

Sinais
Controlados por um motor.

LED/Pares de resistores fotossensíveis
Esses dois componentes constituem um detector que acusa a presença do trem.

Desvios
Controlados por um motor.

ma escala de valores, permitindo que seja usado para medir mudanças de temperatura, distâncias, rotação de objetos, o peso de alguma coisa ou a voltagem de uma bateria. Esse tipo de sinal é processado por um conversor analógico-digital (abreviado por A/D), que converte os valores variáveis do equipamento controlado (os sinais analógicos) em forma digital compreensível ao micro.

Resultado final

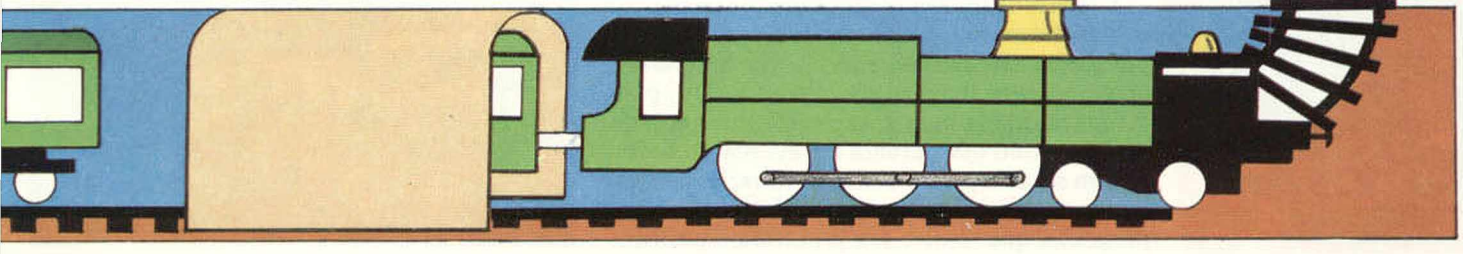
A existência de um dispositivo de realimentação faz grande diferença em tudo o que é controlado pelo computador. Se um motor está impulsionando uma roda, o micro pode calcular o quanto ela girou num dado tempo. Contudo, se existe algum peso sobre a roda, ou se as baterias que alimentam o motor estão quase descarregadas, um sensor óptico pode manter o computador informado dessas variações, notificando-o cada vez que a roda completa uma volta.

Motores elétricos desenvolvidos para serem controlados por computador têm uma espécie de realimentação embutida. O micro envia uma mensagem para que o motor se mova até determinada posição, na qual ele se desliga de modo automático. Há dois motores desse tipo: o gradual e o servomotor. O motor gradual funciona girando continuamente, de modo comum, e pode ser parado em qualquer posição. Já o servomotor só gira um pequeno ângulo — em geral, até 90 graus. Esse tipo de movimento é convertido em deslocamento vertical (puxa-empurra). Para que funcionem com computadores, ambos os motores exigem sistemas especiais de controle, e estes não estão disponíveis para muitos tipos de micro.

Na maioria das vezes, os servomotores são utilizados em braços mecânicos, tipo robô. Existem alguns braços robotizados pequenos que podem ser conectados a microcomputadores, mas são muito caros. É possível construir um braço robotizado com alguns servomotores a um custo mais baixo.

Outros tipos de dispositivos exigem voltagem variável para ser controlados por micros. Por exemplo, um pequeno motor elétrico que roda a diferentes velocidades, dependendo da voltagem a ele aplicada. O oposto de um conversor A/D é um conversor digital-analógico (D/A), que transforma os sinais digitais utilizados pelo computador em voltagens variáveis; por exemplo, para produzir sons, quando ligado a um alto-falante.

Usar um microcomputador para controlar outros equipamentos é como programar o próprio software. Você precisa combinar uma boa idéia, algum conhecimento técnico e muito tempo livre. Muitas vezes os resultados não compensam economicamente, mas o fazer-você-mesmo é muito mais divertido do que comprar pronto.



TRAVESSIA DO DESERTO

Nem todos os jogos têm por objetivo a destruição de seres alienígenas. Alguns, como Travessia do Deserto, exigem bastante raciocínio lógico, um pouco de experimentação e até mesmo alterações no programa.

Neste jogo, você precisa percorrer, dirigindo um caminhão, uma extensão de 1.000 km. Seu itinerário atravessa uma região desértica e inteiramente desabitada. No entanto, a cada 100 km mais ou menos, há uma estação onde você pode armazenar tonéis de combustível em seu veículo. Na base inicial, existe uma quantidade ilimitada de tonéis, mas cada um contém combustível suficiente apenas para o caminhão ir de uma estação até a outra.

A jornada não apresentaria problema se não fosse por um detalhe: o caminhão só pode transportar oito tonéis de cada vez. Desse modo, se você pretende chegar a seu destino, terá de apanhar combustível em vários pontos da rota, o que certamente implicará idas e vindas entre as diversas estações.

Em primeiro lugar, tome cuidado para não ficar sem combustível entre uma estação e outra — afinal, uma longa caminhada sob o sol escaldante do deserto não é perspectiva das mais atraentes. Ao contrário, pode até ser fatal.

O segundo objetivo do jogo é fazer com que toda a travessia seja realizada com o mínimo possível de combustível e percorrendo a menor quilometragem.

Você não encontrará muita dificuldade para solucionar o problema, pois o programa permite que o caminhão transporte oito tonéis de cada vez. Mas ele se tornará mais interessante se for alterado no sentido de exigir maior esforço mental.

O que aconteceria, por exemplo, se o caminhão carregasse apenas quatro ou seis tonéis de cada vez? Para investigar essas alternativas mude o valor da variável M na linha 60 e reconsidere o problema desde o início.

A forma de resolvê-lo será a mesma, como você logo descobrirá. Mas os intervalos entre os depósitos de combustível e as viagens necessárias de um a outro deverão ser alterados. Ou, para complicar ainda mais a questão, tente projetar um algoritmo que garanta a travessia do deserto com segurança, em qualquer situação. Um algoritmo desse tipo permite a elaboração de um programa que resolve seu problema específico.

Hoje, a palavra "cibernética" designa a ciência que estuda comunicações e sistemas de controle não só nos organismos vivos como também em máquinas (os computadores, por exemplo). Originariamente, porém, *kybernetiké* indicava a arte do piloto. Travessia do Deserto permite que você "pilote" com auxílio do micro, transformando dados em ação otimizada.

Uma ferramenta de programação

Este quebra-cabeça de matemática e estratégia ilustra uma técnica valiosa para a solução dos problemas que surgem durante o desenvolvimento de qualquer programa. Primeiro, note como é importante experimentar com as informações disponíveis e testar algumas das possíveis variações. Se tudo correr bem, você acabará descobrindo um padrão geral, que inclui todas as ocorrências específicas. A partir dele, torna-se bem mais fácil projetar um algoritmo e, em seguida, desenvolver um programa abrangente.

Outra maneira de aperfeiçoar o jogo Travessia do Deserto é por meio do acréscimo de gráficos e outros recursos ou, ainda, estabelecendo dificuldades cada vez maiores — como a necessidade de transportar tonéis de água, além dos de combustível. No caso, inventar problemas pode ser tão estimulante quanto tentar resolvê-los.

Variações

O programa do jogo Travessia do Deserto, apresentado na página seguinte, foi escrito em Applesoft e, portanto, pode ser executado em todos os microcomputadores compatíveis com o Apple (inclusive o TK 2000). Se você quiser empregá-lo em algum equipamento da linha Sinclair (TK 85, CP 200 etc.), altere as linhas indicadas a seguir:

```
 110 CLS
 210 REM ( E L I M I N A R )
 220 REM ( E L I M I N A R )
 240 PRINT TAB (X-2);"▄▄";CHR$ (
128+CODE STR$ T);TAB (X-2);"▀ ▀"
 250 REM ( E L I M I N A R )
 320 PRINT "OH...OH... E▀ MELHOR
 VOCE        INICIAR NOVAMENTE."
 810 PRINT "VOCE E ";A;" TONEIS
FICARAM","NO DESERTO."
```

Para executá-lo num TK 90X, além das alterações acima, modifique a linha 240, escrevendo-a do seguinte modo:

```
 240 PRINT TAB (X-2);"▄▄ ";INVER
SE 1,T; INVERSE 0;TAB (X-2);"▀ ▀
```

Programa básico

Linha Apple

Travessia do Deserto

```
10  REM  TRAVESSIA DO DESERTO
20  REM  INICIALIZACAO
30  DIM A(10)
40  LET A(1) = 80
50  LET S = 1
60  LET M = 8
70  LET N = 0
80  LET T = 0
90  REM  VIAGENS
110  HOME
120  PRINT "E)  1  2  3  4  5  6
     7  8  9'10"
130  PRINT "T) ";
140  FOR I = 1 TO 10
150  LET A$ =  STR$ (A(I))
160  IF  LEN (A$) >  = 2 THEN  GOTO
     190
170  LET A$ = " " + A$
180  GOTO 160
190  PRINT A$;" ";
200  NEXT I
210  PRINT
220  PRINT
230  LET X = 4 + (S - 1) * 3
240  PRINT "ESTACAO:";S
250  PRINT "TONEIS NO CAMINHAO:";
     T
260  IF S <  > 10 THEN  GOTO 300
270  PRINT
280  PRINT "VOCE CONSEGUIU ..."
290  GOTO 740
300  IF  NOT (T = 0 AND A(S) = 0)
      THEN  GOTO 340
310  PRINT
320  PRINT "OH...OH... E MELHOR V
     OCE INICIAR": PRINT "NOVAMEN
     TE."
330  GOTO 740
340  PRINT
350  PRINT "SUAS OPCOES SAO:"
360  PRINT
370  IF T > 1 THEN  PRINT "R RETI
     RAR TONEIS DO CAMINHAO"
380  IF A(S) > 0 THEN  PRINT "P P
     EGAR TONEIS DE COMBUSTIVEL"
390  IF T > 0 THEN  PRINT "A AVAN
     CAR UM ESTAGIO"
400  IF T > 0 AND S > 1 THEN  PRINT
     "V VOLTAR UM ESTAGIO"
410  PRINT
420  PRINT "QUAL E A ESCOLHA"
430  INPUT A$
440  IF T > 1 AND A$ = "R" THEN  GOTO
     570
450  IF A(S) > 0 AND A$ = "P" THEN
      GOTO 650
460  IF  NOT (T > 0 AND A$ = "A")
      THEN  GOTO 510
470  LET S = S + 1
480  LET T = T - 1
490  LET N = N + 1
500  GOTO 110
510  IF  NOT (T > 0 AND S > 1 AND
     A$ = "V") THEN  GOTO 110
520  LET S = S - 1
530  LET T = T - 1
540  LET N = N + 1
550  GOTO 110
560  REM  DESCARREGAR TONEIS
570  PRINT
580  PRINT "QUANTOS";
590  INPUT A
600  IF (A > T) OR (A <  > INT (
     A)) OR (A < 0) THEN  GOTO 59
     0
610  LET A(S) = A(S) + A
620  LET T = T - A
630  GOTO 110
640  REM  CARREGAR TONEIS
650  PRINT
660  PRINT "QUANTOS";
670  INPUT A
680  IF A > A(S) OR A <  > INT (
     A) OR A < 0 THEN  GOTO 670
690  IF A + T > M THEN  GOTO 670
700  LET T = T + A
710  LET A(S) = A(S) - A
720  GOTO 110
730  REM  FIM DO JOGO
740  PRINT
750  PRINT N;" TONEIS FORAM GASTO
     S."
760  PRINT T;" FICARAM NO CAMINHA
     O."
770  LET A = 0
780  FOR I = 2 TO 9
790  LET A = A + A(I)
800  NEXT I
810  PRINT "VOCE E ";A;" TONEIS F
     ICARAM NO DESERTO."
820  PRINT
830  PRINT "COMANDE ""RUN"" PARA
     TENTAR ","NOVAMENTE."
```

PRINTER/ PLOTTERS

O preço desses periféricos ultrapassa o orçamento da maioria dos usuários de micros, mas existem, no mercado internacional, modelos acessíveis e adequados ao uso pessoal.

Equipamentos complexos, os plotters são empregados para produzir gráficos detalhados em cores. Funcionam ainda como impressora, produzindo textos, e por isso são chamados também de "printer/plotters".

Dentro da unidade, há um tambor giratório com quatro pequenas canetas — esferográficas ou hidrográficas — de cores vermelha, azul, verde e preta. Para desenhar uma linha, o tambor gira, selecionando a cor desejada. Em contato com o papel, a caneta traça linhas horizontais (movimentos laterais do tambor) e verticais (movimentos do papel para cima e para baixo).

Os caracteres de texto são produzidos quase da mesma forma que os gráficos. O printer/plotter armazena os padrões das letras e outros caracteres em sua própria memória. Quando recebe um sinal do computador, a unidade apenas encontra o caractere requisitado na memória e o desenha como se fosse um padrão. A qualidade de impressão é muito boa, melhor que a da maioria das impressoras matriciais.

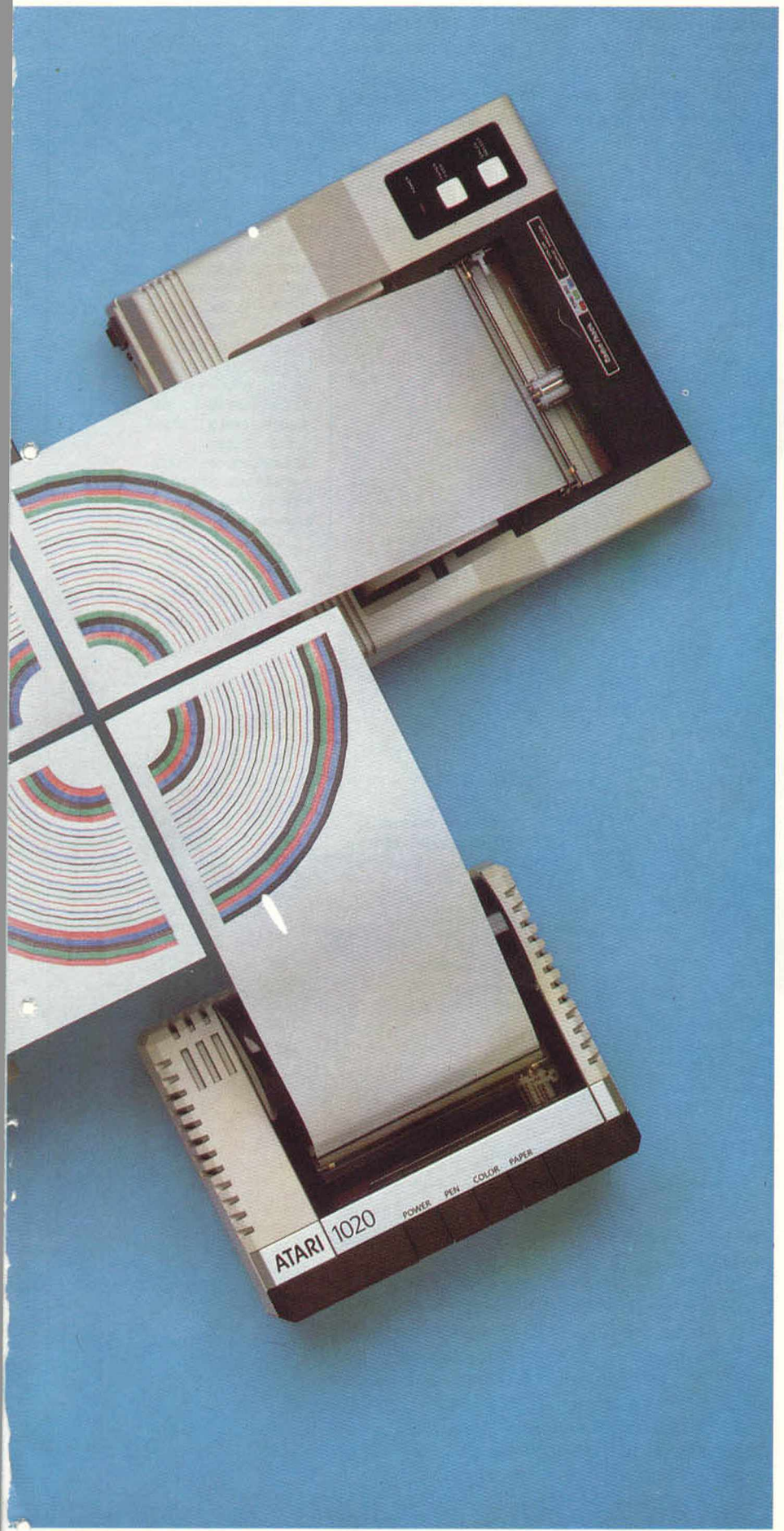
Em modo texto, o printer/plotter funciona da mesma forma que qualquer outra impressora. Embora o papel empregado por alguns modelos tenha apenas 11,5 cm de largura, a unidade produz quarenta ou oitenta caracteres por linha. Com papel estreito, um texto com oitenta caracteres resulta em letras pequenas, mas bastante nítidas.

É possível imprimir textos em qualquer das quatro cores, a dez caracteres por segundo. Essa velocidade mostra-se baixa, em relação às impressoras matriciais, mas aproxima-se à velocidade das impressoras tipo margarida.

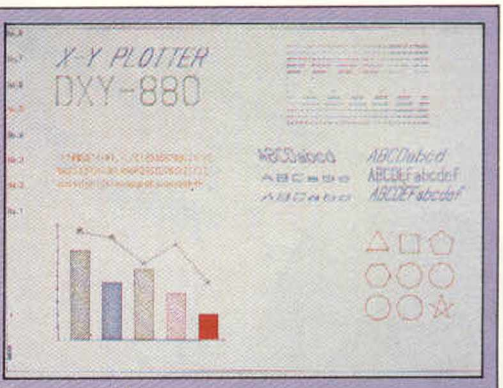
Para produzir gráficos, o printer/plotter é chaveado para o modo gráfico, no qual produz linhas, curvas e letras grandes. Caracteres enviados para a unidade nesse modo não são impressos, pois no caso ela os interpreta como comandos. Quando se pretende traçar uma linha, precisa-se de comando, que em BASIC seria:

```
LPRINT "D 0,100,100,100,100,0,0,0"
```

A letra D é a instrução para traçar uma linha, e os números que se seguem dão a posição dos pontos pelos quais a linha vai passar. No caso,



Opção profissional
Nos plotters convencionais são utilizadas folhas de papel avulsas, adequadas a cada aplicação.



Recursos variados
Além de diagramas e gráficos, os plotters também podem funcionar como impressoras, "desenhando" os caracteres.

O printer/plotter DXY 880, com o qual foi produzido o desenho *acima*, é da fábrica japonesa Roland. Utiliza oito canetas e papel de 44 x 29 cm. Seu chip de ROM incorpora várias rotinas de desenho, que podem ser chamadas pelo programa que está rodando no micro: DRAW, CIRCLE, MOVE, HOME, LINE, SCALE, HATCHING etc. *À esquerda*, trabalhos com outros printers/plotters do mercado internacional.

supondo a cabeça de impressão localizada no início das coordenadas (0,0), o plotter desenhará um quadrado.

Considera-se a largura do papel dividida em 480 "espaços" horizontais, usados para posicionar o tambor. O papel utilizado pela unidade forma rolos de vários metros de comprimento. Isso pode dar a falsa impressão de não haver limite para a extensão vertical do desenho. Na verdade, porém, o deslizamento do papel pode resultar em linhas que não se unem conforme o desejado. Por isso, o printer/plotter impede que o papel seja puxado de volta além de certa distância.

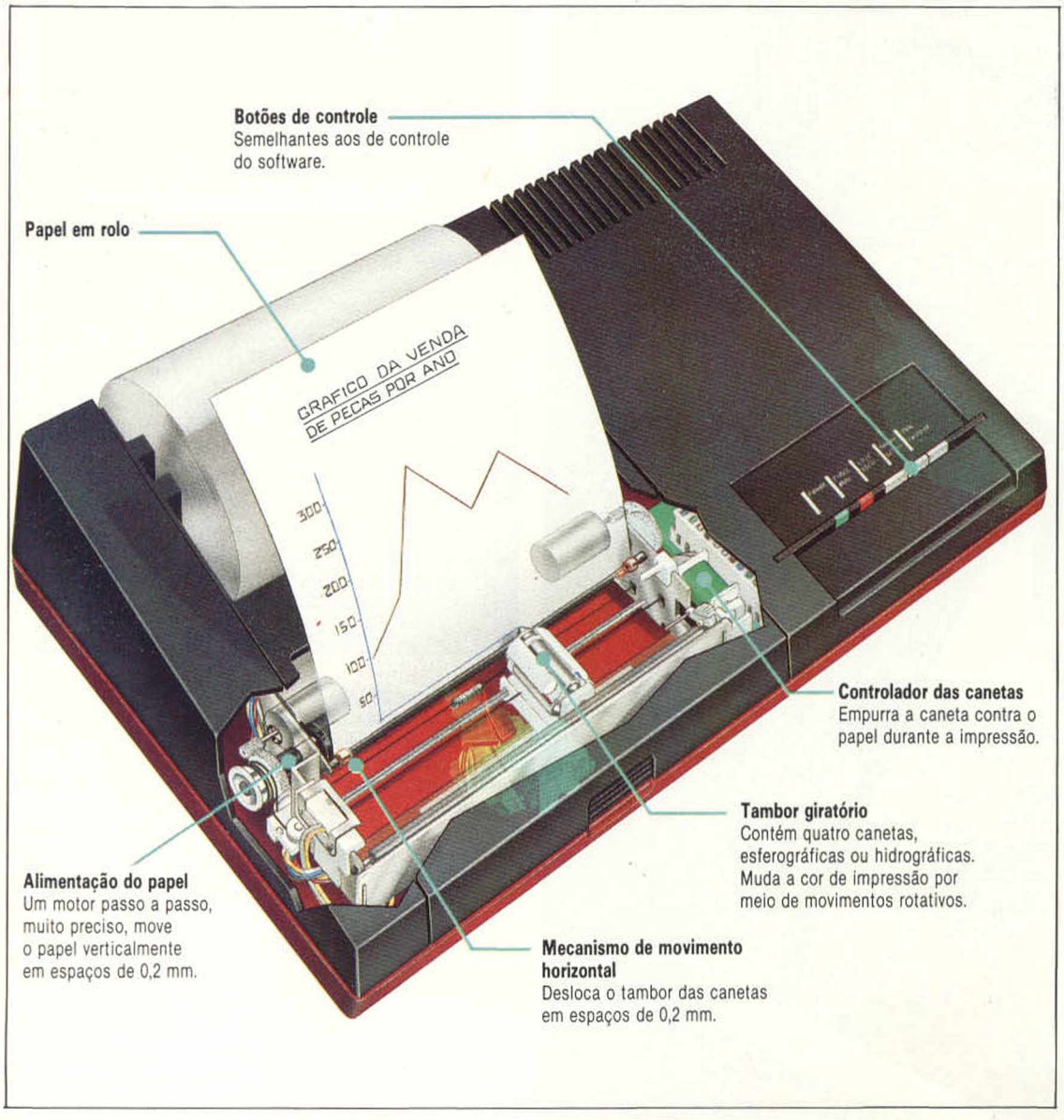
É possível produzir gráficos complexos no printer/plotter, mas a programação necessária demora muito. Não há forma simples de copiar uma imagem diretamente da tela do computador, como fazem alguns aplicativos para impressoras matriciais. Desenhar uma linha é muito simples, mas a ausência de comandos como PAINT e FILL significa que, para preencher uma forma com cores, precisamos traçar muitas linhas contíguas, processo que demora e consome muita tinta.

Também não se pode empregar toda a largura do papel — uma margem deve ser deixada de cada lado.

O equipamento é mais adequado à produção de gráficos, dispondo de um comando que dessenha automaticamente os eixos, já com as unidades requeridas marcadas nos intervalos apropriados.

Em modo texto, a unidade produz dois tamanhos de letras. Contudo, utilizada em modo gráfico, obtêm-se tamanhos de caracteres muito maiores, sendo possível imprimir no sentido lateral. Portanto, imprimem-se textos longos no sentido do comprimento do papel.

As canetas ficam sem tinta rapidamente, e podem ressecar quando deixadas na cabeça de impressão. Cada vez que se liga o equipamento, ele desenha um quadrado de cada cor, num teste automático das canetas.



Papel e tambor
O printer/plotter tem uma resolução de aproximadamente 500 x 2.000 pixels no modo gráfico e vários tamanhos de caracteres. A cor é escolhida pela rotação do tambor das canetas. Obtém-se o texto ou o desenho pelos movimentos horizontal do tambor (da esquerda para a direita) e vertical do papel.
Os desenhos são de alta precisão, mas o movimento repetido do papel e do tambor provoca leve perda de registro.

BBC MODELO B

Boa qualidade técnica e grande disponibilidade de software, apoiadas no eficiente marketing da BBC, tornaram esse microcomputador muito popular na Inglaterra.

O BBC Modelo B é fabricado pela Acorn Computers, em Cambridge, e comercializado pela BBC (British Broadcasting Corporation, a rádio e tevê estatal britânica). Havia dois tipos, mas o primeiro deles, o Modelo A, mais barato e menos sofisticado, foi retirado de linha.

O Modelo B teve boa aceitação nas escolas e conta com o aval do governo como material pedagógico. Considerável software educacional foi desenvolvido para esse equipamento — desde linguagens de programação até pacotes para aprendizagem apoiada em computador.

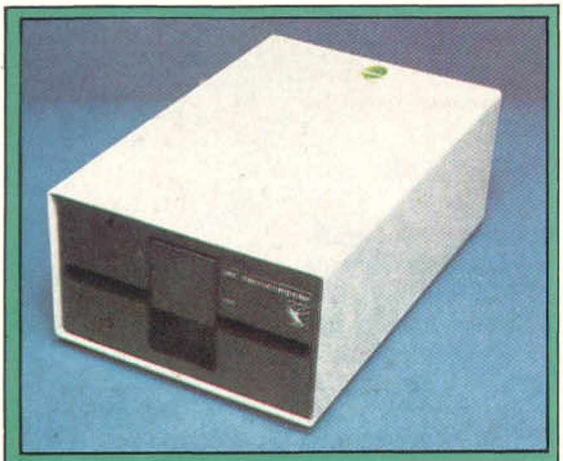
Os recursos técnicos do BBC Modelo B ainda são considerados excelentes, apesar da grande variedade de novas máquinas lançadas mais recentemente no mercado inglês. A linguagem de programação BBC BASIC, em particular, dispõe de amplo leque de comandos para operar com funções especiais, o que facilita a tarefa de desenvolver e editar programas.

O equipamento possibilita oito modos gráficos distintos. Isso significa que o usuário pode escolher a resolução entre baixa, média e alta, apesar de, no último caso, o número de cores disponíveis ficar limitado. O máximo de resolução que se consegue é de 640 x 256. Muitos usuários optam por ligar a CPU a um televisor comum, mas recomenda-se um monitor de vídeo dedicado, a fim de se obterem os melhores resultados dos gráficos do Modelo B. Há comandos para desenhar linhas, círculos e construir uma série de imagens na tela.

Seu design, em função da fonte de alimentação interna, apresenta-se bem-acabado e compacto, mas as interfaces — na parte posterior e na inferior do gabinete — são mais numerosas do que em outros equipamentos. Isso significa que grande variedade de dispositivos para expansão pode ser encontrada, inclusive várias marcas de unidades de disco não fabricadas especialmente para o BBC.

Além das interfaces para unidades de disco, impressora e um dispositivo analógico para controle de equipamentos de laboratório ou instrumentos de medida, o Modelo B pode facilmente trabalhar em rede. Isso o torna ideal para salas de aula, onde vários usuários podem partilhar a mesma unidade de disco ou impressora.

Teclado funcional
Um dos pontos fortes do BBC Modelo B. Com bom layout, dispõe de vários recursos, além de acabamento sólido. Pessoas acostumadas com teclado de máquinas de escrever comuns sentem-se à vontade.
As quatro teclas com setas, ao lado direito, destinam-se a movimentar o cursor na tela, para editar textos ou programas. As dez teclas vermelhas, de funções programáveis, são especialmente úteis para programas educacionais, pois o usuário tem apenas de escolher a resposta certa em dez alternativas.
Uma característica interessante é a inclusão de três LEDs para indicar se o motor do cassete está funcionando e se as teclas [Shift] ou a trava das maiúsculas foram acionadas.



Unidade de disco
A unidade de disco do BBC Modelo B é um acessório muito atraente, mas custa caro e armazena apenas 100 Kbytes. Existem outras marcas e a preços inferiores que devem ser levados em consideração.

Entrada analógica
Possibilita ao computador ler sinais analógicos — por exemplo, de um sensor de calor. Mais usada em laboratórios ou trabalhos experimentais.

Controlador de vídeo
Este chip obtém dados provenientes da RAM e os converte em sinais de vídeo.

Chips estrangeiros
Observando cuidadosamente estes chips, você perceberá que a fabricação de microprocessadores é uma questão internacional. O BBC Modelo B contém chips fabricados em Malásia, Japão, Portugal, Escócia e Estados Unidos.

Processador de interfaces
Adaptadores versáteis de interface como este 6522, de tecnologia MOS, cuidam da ligação com dispositivos externos.

Saídas para periféricos
Parte inferior do BBC Modelo B, vendo-se as tomadas onde se conectam os periféricos.

Porta para impressora
Liga qualquer impressora paralela.

Porta digital
Para testes com dispositivos digitais e circuitos lógicos feitos pelo próprio usuário.

SCOPUS

HÁ 10 ANOS DESENVOLVENDO TECNOLOGIA BRASILEIRA

No Brasil, a era da informática começou recentemente. Mas não para a SCOPUS que, desde 1975, trabalha em projetos, produções, criações e fabricações das mais avançadas, destacando-se no mercado porque aposta no talento e capacidade de técnicos, engenheiros e especialistas brasileiros. Não mede esforços. Sempre à procura de soluções modernas e criativas, desenvolve o que há de mais arrojado em matéria de informática. Sem copiar o que é feito fora. Conheça agora algumas das atuais conquistas tecnológicas da SCOPUS.

Tecnologia e sofistic

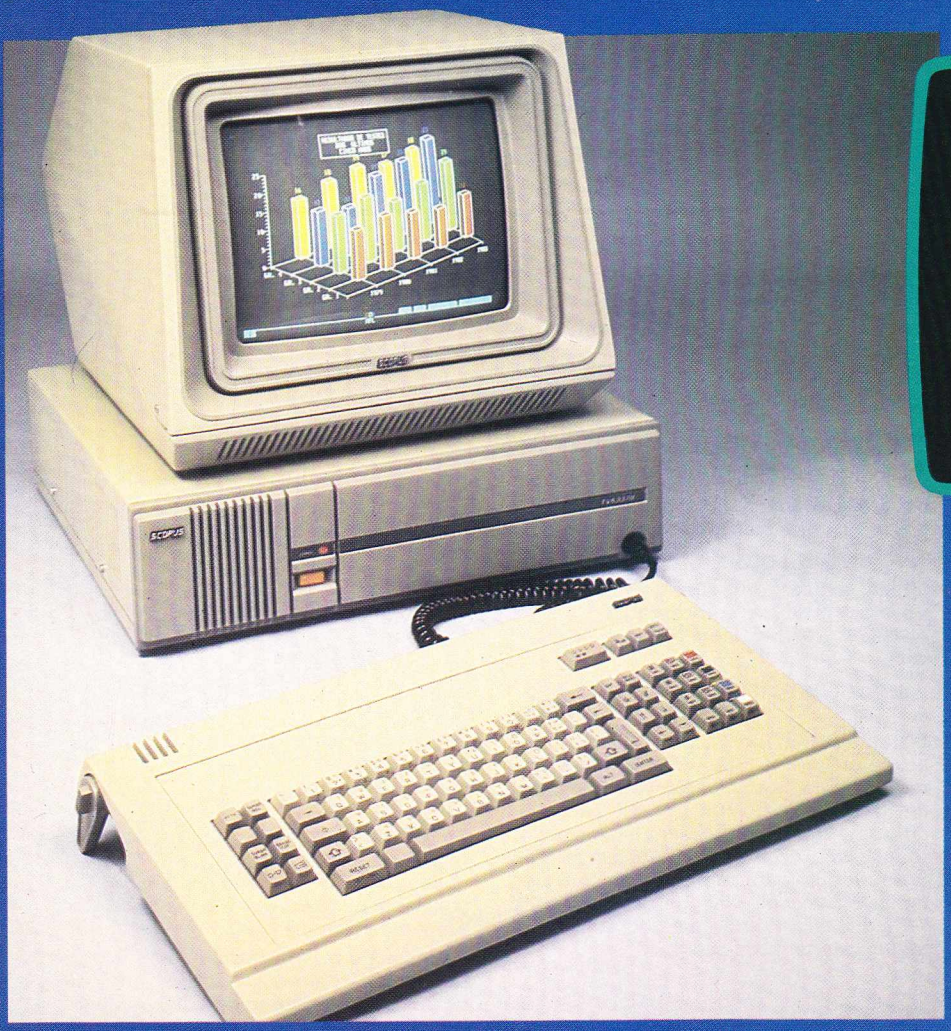
NEXUS 1684

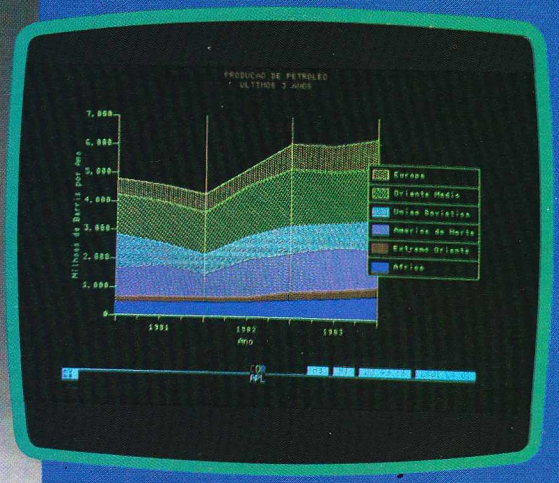
Como o mais recente integrante da família NEXUS 1600 de microcomputadores, o modelo 1684 traz uma importante novidade: a integração da unidade de disco magnético "Winchester", capaz de armazenar 10 milhões de caracteres (10 Mbytes). Dirigido a aplicações profissionais, o NEXUS pode ser utilizado para diversas operações, tais como análise financeira, análise de orçamento, balancete, entre outras tarefas gerenciais e administrativas. Ele se integra também a vários sistemas de grande porte, como computadores IBM, Burroughs ou Honeywell. Oferece características exclusivas, como o "clock" de CPU de 8 MHz e o tratamento de vídeo sem cintilação. Por trabalhar junto com a unidade Winchester, foi possível obter-se uma grande redução de custos além de um melhor aproveitamento do espaço físico.

TVA 3379

Com um hardware sofisticadíssimo e tecnologia (brasileira, é claro) das mais avançadas, o terminal de vídeo TVA 3379 tem como proposta principal facilitar a interpretação de dados fornecidos pelo computador através de gráficos a cores. É o único terminal desse tipo disponível para sistemas IBM existente no mercado. Usando o recurso das cores, o TVA 3379 facilita a leitura e inter-

ção em três versões.





pretação de qualquer informação, o que antes só poderia ser obtido através de tabelas numéricas. É a SCOPUS atenta a todo avanço e tendência do mercado mundial, facilitando a relação entre o homem e a máquina. Apenas usando o talento, o esforço e a criatividade de centenas de engenheiros brasileiros.

PC 2140

O mais novo e econômico "tradutor de linguagens", criado e desenvolvido com tecnologia totalmente nacional, o processador de comunicação PC 2140 permite a conexão simultânea de 32 equipamentos (terminais ou microcomputadores) com um computador de grande porte, através de uma única linha BSC-3. Isto permite o acesso ao banco de dados dos sistemas de grande porte, como IBM ou Burroughs, possibilitando ao usuário levantar importantes informações disponíveis no computador, entre elas, saldos de conta bancária, consultas ao pregão da Bolsa de Valores e verificação do estoque. O PC 2140 pode ser ligado a Unidades de Controle de Transmissão IBM 270X, 370X e ICA. A ligação dos equipamentos ao PC 2140 deve ser direta ou via modem, podendo-se também utilizar o modem de resposta automática.

Qualidade e assistência. A Scopus garante.

A avançada tecnologia do NEXUS 1600, PC 2140 e TVA 3379 já garante ao consumidor alto grau de qualidade e confiabilidade. Mas não é só isso que a SCOPUS oferece. Todos os serviços de manutenção são prestados por equipes de profissionais de alto nível técnico, em constante treinamento, que atendem rapidamente a qualquer chamado. Sempre investindo no aperfeiçoamento de sua estrutura de atendimento, a SCOPUS garante a mais elevada disponibilidade a seus equipamentos e a certeza de uma assistência técnica permanente.

DADOS TÉCNICOS

NEXUS 1600

UNIDADE DE SISTEMA

Placa de sistema: Microprocessador Intel 8088 (4,77 ou 8 MHz) • 64 Kbytes de memória EPROM • 256 Kbytes de memória RAM • 2 interfaces de comunicação serial: assíncrona e síncrona/assíncrona, padrão RS-232C, em velocidades até 19200 bauds • 5 conectores para placas de expansão.

Placa controladora de disco: controla duas unidades de disco flexível de 5 1/4", possui relógio-calendário de tempo real não volátil e duas interfaces para "joy-sticks".

Placa controladora de vídeo: modo alfanumérico: telas de 40x25 ou 80x25 caracteres, 16 cores • modo gráfico: telas de 320x200 pontos a 4 cores ou 640x200 pontos monocromáticas, interface para "light-pen".

Placa controladora de Winchester: controla a unidade de disco rígido tipo Winchester.

Unidade de disco flexível de 5 1/4": um acionador com cabeça dupla, densidade dupla, com capacidade de 320 Kbytes.

Unidade de disco rígido tipo Winchester: um acionador com capacidade de 10 Mbytes, slim.

MONITOR DE VÍDEO

Monocromático, 12", fósforo verde, interface padrão RGB, média resolução e configuração multinível, reproduzindo a informação da cor em tons de intensidades variáveis. Opcionalmente, pode-se configurar o Nexus 1600 com um monitor de vídeo cromático, 14", média resolução.

TECLADO

Tipo máquina de escrever, 86 teclas.

SOFTWARE BÁSICO

O Sistema Operacional SISNE, desenvolvido pela Scopus, emula o sistema MS-DOS 1.1 do microcomputador PC-IBM e contém todos os utilitários compatíveis com os utilitários do MS-DOS 1.1. Possui uma interface homem-máquina bastante poderosa e gerencia, com máxima eficiência, os recursos físicos do equipamento.

MÓDULOS DE EXPANSÃO

N1664: expansão de memória RAM (448 Kbytes) • **N1667**: co-processador CP-8087 • **N1669**: interface paralela para impressora.

MÓDULOS PARA COMUNICAÇÃO DE DADOS

Sistema Incox: interface coaxial para conexão a redes IBM 3270 • **NX-3270**: módulo para comunicação com sistemas IBM sob protocolo BSC-3 • **NX-830**: módulo para comunicação com sistemas Burroughs sob protocolo Poll-Select • **NX-VIP**: módulo para comunicação com sistemas Honeywell Bull sob protocolo VIP 7700 • **NX-Z**: módulo para comunicação assíncrona serial de dados.

TVA 3379

MÓDULOS

Unidade de vídeo: apresenta tela útil de 1920 ou 2560 posições, respectivamente para os modelos 2 ou 3. Possui duas linhas adicionais de informações no vídeo para mensagens do sistema e de erros, além de possibilitar a apresentação de cronômetro e indicador de linha e coluna. O monitor é cromático, de alta resolução. O cinescópio, de 14 polegadas, pode ser configurado para que se apague automaticamente, evitando a queima desnecessária do fósforo.

Unidade lógica: comporta as placas de controle do terminal, de geração de caracteres plano triplo, e as interfaces de comunicação com o vídeo, unidade controladora e teclado.

Teclado: comunica-se com a unidade lógica de forma serial e permite a conexão de um dispositivo auxiliar de entrada. Está disponível em quatro opções: APL Português, APL Inglês, "Typewriter" Português e "Typewriter" Inglês, todas com bloco destacado de teclas de cor e de símbolos programáveis.

PC 2140

CARACTERÍSTICAS FUNCIONAIS

Interface de comunicação com terminais ou micros: serial, assíncrona, código ASCII, full duplex, padrão elétrico RS-232C.

Placa interface de comunicação: cada placa permite que até 8 dispositivos sejam conectados ao PC 2140 com seleção de velocidades independentes.

Conexão de modem: a interface de comunicação com os terminais ou micros permite a ligação de modem, podendo ser constante ou por meio de linhas discadas, através de centrais telefônicas.

Interface de comunicação com o computador: serial, síncrona, sob o protocolo BSC-3, padrão elétrico RS-232C, em velocidades até 9600 bps. Emula as unidades de controle tipo IBM 3271 ou 3274 com 32 terminais IBM 3277 ou 3278, modelo 1.

Conversão de códigos: adotados pelos terminais (ASCII) em códigos adotados pelo computador (EBCDIC).

Execução de comandos: permitem ao computador ativar funções como, por exemplo, estabelecer conexões com os terminais, desfazer conexões, transmitir textos e informar o estado de ligações.

Verificação de paridade: em todas as mensagens com o computador, o PC 2140 gera e verifica os caracteres de controle de bloco (CRC-16), pedindo retransmissão automática em caso de erro.

MODELOS

O PC 2140 está disponível em quatro opções, que diferem apenas em relação ao número de placas de controle dos terminais.

PC 2140/8: controla até 8 terminais (1 placa).

PC 2140/16: controla até 16 terminais (2 placas)

PC 2140/24: controla até 24 terminais (3 placas)

PC 2140/32: controla até 32 terminais (4 placas)

CENTROS DE ASSISTÊNCIA TÉCNICA

BELÉM: Tel.: (091) 228-0011 - ramal 242
BELO HORIZONTE: Tel.: (031) 224-8457
BRASÍLIA: Tels.: (061) 224-9856 e 224-9835
CAMPINAS: Tel.: (0192) 42-7333 - BIP: 657
CUIABÁ: Tels.: (065) 322-0542 321-5792 - BIP: 139
CURITIBA: Tel.: (041) 262-5211 - BIP: 150
FORTALEZA: Tel.: (085) 224-7999 - BIP: DA2
LONDRINA: Tel.: (0432) 22-0558
PORTO ALEGRE: Tel.: (0512) 32-1200 - BIP: 798
RECIFE: Tel.: (081) 221-3100 - BIP: 765
RIBEIRÃO PRETO: Tels.: (016) 636-3967 634-3331/3509 - BIP: 76B
RIO DE JANEIRO: Tels.: (021) 240-4925 e 240-6114
SALVADOR: Tel.: (071) 233-4117 - BIP: D12
SÃO JOSÉ DOS CAMPOS: Tels.: (0123) 22-8247 21-2004 - BIP: 536
SÃO PAULO: Tel.: (011) 255-7744

SCOPUS tecnologia s.a.

Filiada à ABICOMP

As especificações contidas neste folheto podem ser alteradas pelo fabricante sem aviso prévio.

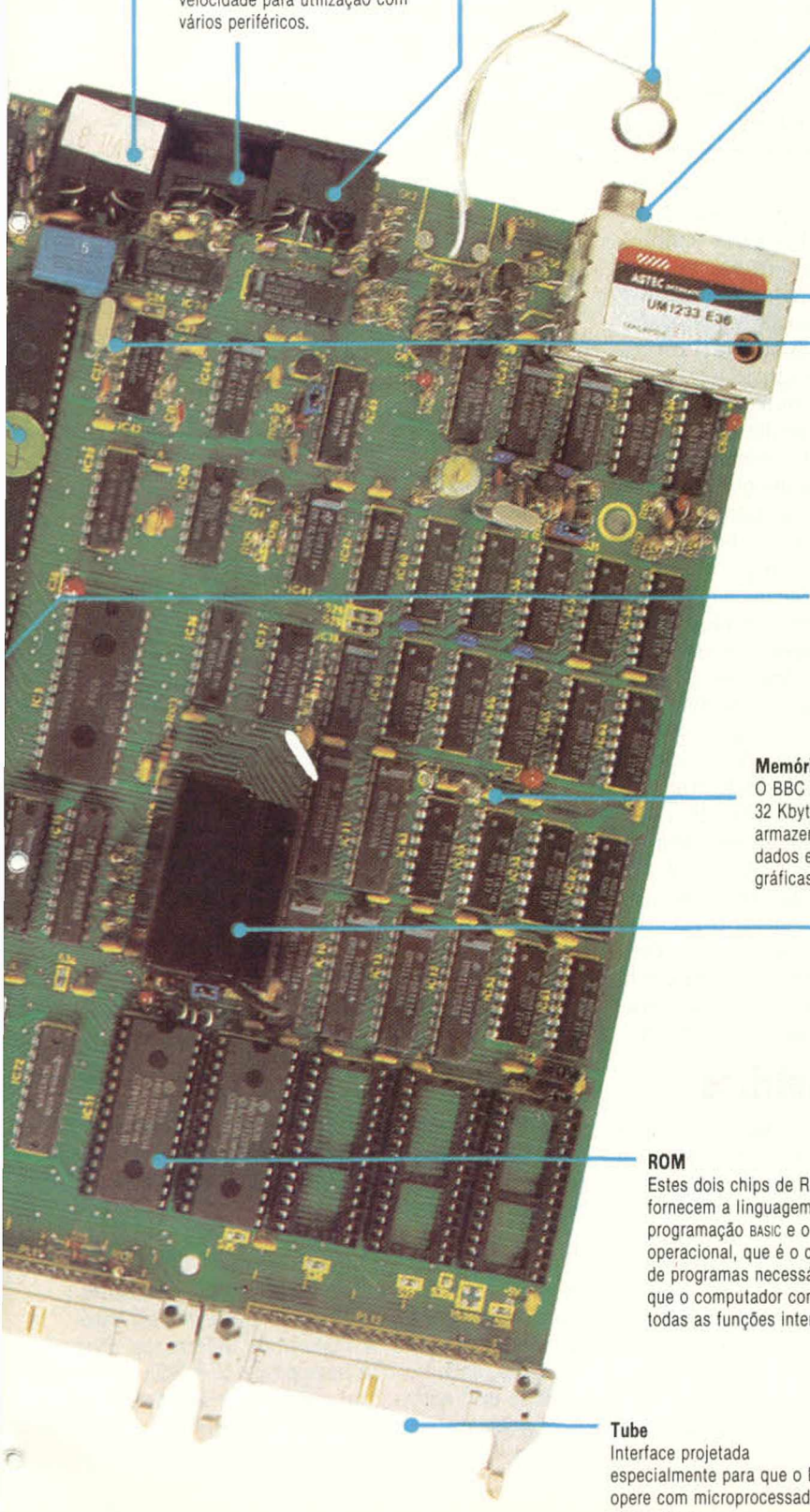
Interface para cassete
Programas em gravador cassete comum são guardados de duas formas: em velocidade máxima ou com maior fidelidade da gravação.

Saída RGB
Fornece sinais separados para os componentes vermelho, verde e azul do sinal do vídeo. Controla um monitor em cores de alta qualidade.

Saída para televisor
Para ligar à antena do aparelho.

Saída para vídeo composto
Para utilização com um monitor monocromático ou colorido.

Entrada RS232C
Interface serial de alta velocidade para utilização com vários periféricos.



Modulador
Conduz o sinal de cores do controlador de vídeo e o converte em saída adequada para o televisor.

Cristal de quartzo
Produz os pulsos de clock, sincronizando todas as operações.

Microprocessador
O 6502 de tecnologia MOS executa todo o trabalho de processamento.

Memória para o usuário
O BBC Modelo B contém 32 Kbytes de RAM para armazenamento de programas, dados e apresentações gráficas.

ULA (Uncommitted Logic Array)
Matriz de lógica não determinada. Executa o trabalho que em outros computadores exigiria dezenas de componentes. A peça de metal na parte superior age como isolante térmico e protege o chip contra superaquecimento e radiofreqüência.

ROM
Estes dois chips de ROM fornecem a linguagem de programação BASIC e o sistema operacional, que é o conjunto de programas necessários para que o computador controle todas as funções internas.

Tube
Interface projetada especialmente para que o BBC opere com microprocessadores alternativos.

BBC Modelo B

DIMENSÕES
409 x 358 x 78 mm.

PESO
3,7 kg.

MICROPROCESSADOR
6502A.

CLOCK
2 MHz.

MEMÓRIA
32 Kbytes de RAM; 32 Kbytes de ROM, incluindo BASIC e sofisticado sistema operacional.

VÍDEO
Oito modos gráficos, com ampla escolha de displays. Área máxima de texto: 32 linhas de 80 caracteres. Resolução gráfica de 640 x 256 pixels. Em baixa resolução, mais de dezesseis cores.

TECLADO
Tipo máquina de escrever, com 74 teclas, incluindo dez de funções programáveis e quatro para controle do cursor.

LINGUAGENS
BASIC, LISP, FORTH, LOGO.

INTERFACES
Televisor, monitores monocromáticos e em cores, unidade de disco, impressora, entrada analógica, porta digital e Tube.

DOCUMENTAÇÃO
O guia do usuário parece presumir que seus leitores já têm familiaridade com computadores. Extensos capítulos são dedicados à utilização especializada de programas que controlam gráficos sofisticados, som e dispositivos de entrada/saída. Está incluída uma explicação detalhada e abrangente do funcionamento e programação do microprocessador 6502.

DIMENSÕES MATRICIAIS

Graças à flexibilidade de suas estruturas de dados, o PASCAL não depende tanto das matrizes quanto outras linguagens. Neste artigo, veremos seus recursos de compactação e dimensionamento de matrizes.

Em muitas linguagens de programação, considera-se a matriz o método mais prático para representar dados do mundo real — por exemplo, listas e tabelas — numa estrutura de dados compreensível pelo computador. Outra vantagem é que ela permite acessar, para processamento, elementos isolados. No entanto, deve-se principalmente ao hábito seu emprego quase universal. Além das matrizes, não havia outro método de estruturação de dados quando se desenvolveram as primeiras linguagens de alto nível. O FORTRAN, por exemplo, possuía somente números complexos e matrizes; e seu descendente, o BASIC, omitia os números complexos. Por esse motivo, o loop FOR-NEXT é a única estrutura definida de iteração.

Isso não constituía problema quando o FORTRAN era empregado apenas para executar operações com matrizes em cada elemento de uma tabela, ou quando empregava-se o BASIC para familiarizar os estudantes com a rigorosa programação do FORTRAN. Já vimos alguns exemplos da extraordinária flexibilidade de controle que os loops condicionais (WHILE e REPEAT) oferecem ao PASCAL. E — talvez ainda mais importante — começamos agora a compreender o enorme poder e a facilidade de expressão que oferecem algumas das outras estruturas de dados do PASCAL, como conjuntos e registros. Desse modo, o predomínio das matrizes como uma "ferramenta de dados" universal para programação é consideravelmente reduzido no PASCAL.

Versatilidade das matrizes

Embora a maioria dos programadores sinta-se em terreno familiar ao lidar com matrizes no PASCAL, há dois pontos importantes que devem ser ressaltados. Como essa linguagem utiliza vários métodos de estruturação de dados, muitos dos problemas de programação são resolvidos de maneira mais eficiente por meio de outras estruturas. Em segundo lugar, as matrizes não requerem qualquer limitação quanto à complexidade estrutural de seus elementos, de modo que se pode obter uma flexibilidade muito maior.

A definição de um tipo de matriz exige a reserva de memória para um tamanho determinado de matriz e define tanto o tipo base de seu índice (indexador), como o tipo de cada componente. Assim, por exemplo,

```
TYPE
  CONTACARATER = ARRAY ['A' .. 'Z'] OF
    INTEGER;
VAR
  LISTA : CONTACARATER;
```

reserva espaço de armazenamento para 26 valores integer, indicados pela especificação do identificador da matriz, seguido por uma expressão de indexação entre colchetes. Os colchetes são sempre usados com matrizes, como no BASIC original (o uso subseqüente dos parênteses foi adotado em função de alguns conjuntos de caracteres incompletos, que não possuíam colchetes). Para acessar um determinado valor integer na lista do exemplo, escrevemos

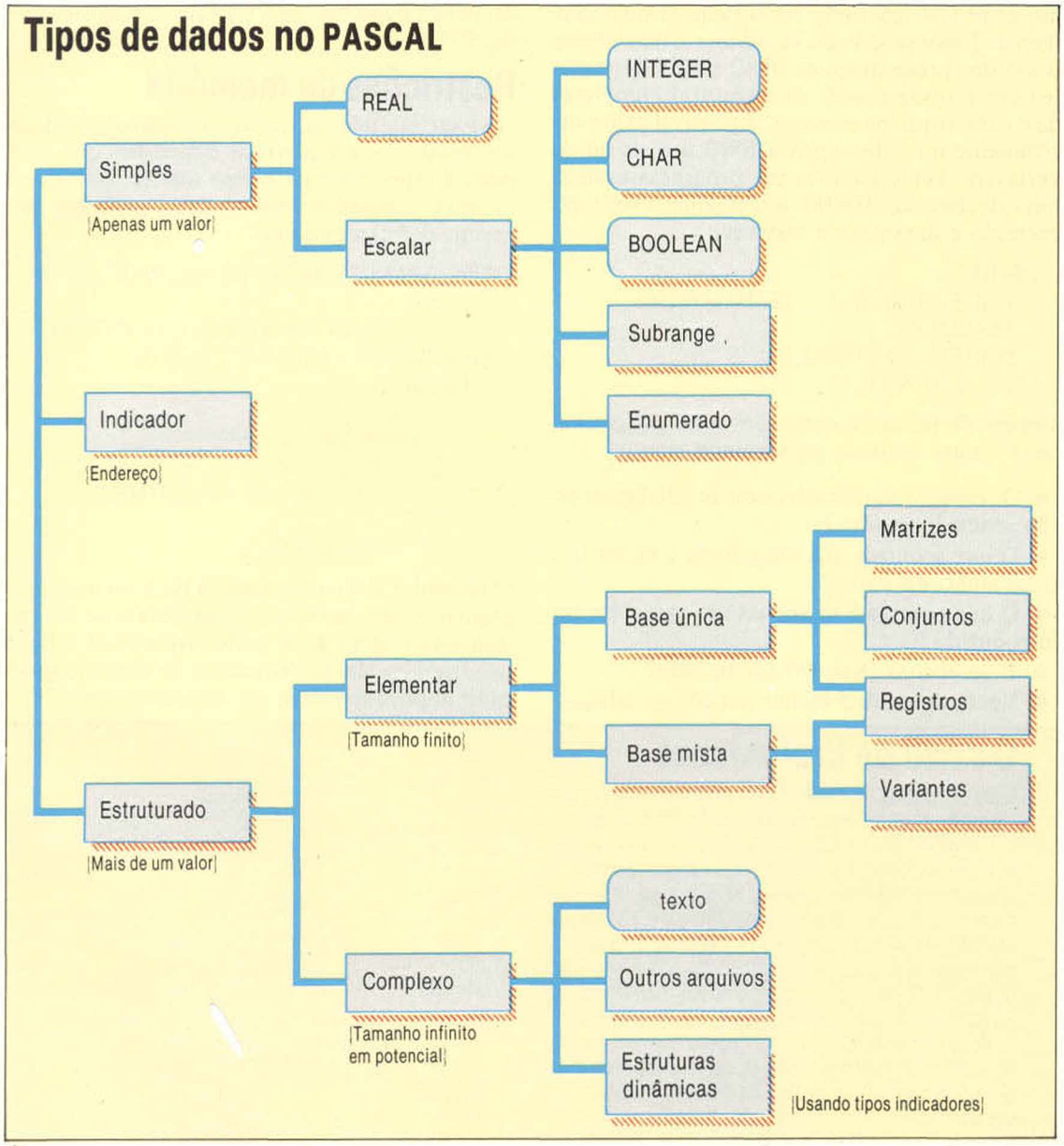
```
LISTA ['M'] ou LISTA [PRED (SIMBOLO)]
```

No segundo exemplo, a expressão PRED (SIMBOLO) deve resultar num valor CHAR na faixa de 'A' a 'Z'. Emprega-se qualquer tipo escalar verdadeiro para indexar dimensões matriciais, mas não tipos reais ou estruturados. Isso garante a inexistência de referências absurdas como, por exemplo, LISTA ['SEGUNDO'] ou FLAG [3,75]. Com a definição de tipo exemplificada acima, inicializamos cada elemento de CONTADOR em zero numa matriz de tipo CONTACARATER:

```
VAR
  LETRA : CHAR;
  CONTADOR : CONTACARATER;
BEGIN
  FOR LETRA := 'A' TO 'Z' DO
    CONTADOR [LETRA] := 0
```

e assim por diante. Seriam ilegítimos, para essa estrutura, tanto CONTADOR [1], pois o tipo de índice está errado, como CONTADOR [A], que não existe. Assim, sempre será um bom procedimento definir o tipo do identificador para as variáveis de indexação. De qualquer forma, são invariavelmente necessários para definição de procedimentos e funções.

Se essas faixas estiverem definidas por valores introduzidos em uma definição CONST, será possível reformular todo o programa — modificando-se apenas uma linha — para que ele opere com estruturas de tamanhos diferentes. Por exemplo, embora não haja o tipo STRING pré-declarado no PASCAL, um modo de criá-lo (embora não necessariamente o melhor) seria



Típico demais!
A rigorosa estruturação de dados no PASCAL obriga a uma abordagem mais metódica na elaboração de um programa. Apenas dois, dentre os três tipos de dados mostrados no esquema, subdividem-se em outras categorias. Mas a variedade possível de dados simples e estruturados exige definições estritas de todos os dados do programa. Esse esforço adicional resulta em soluções mais elegantes para os problemas de programação do que as obtidas em outras linguagens.

```
CONST
  MAXTAMANHO = 25;
TYPE
  TAMSTRING = 1 .. MAXTAMANHO;
  STRING = PACKED ARRAY [TAMSTRING] OF
    CHAR;
```

Note que a palavra reservada PACKED precede qualquer palavra reservada de tipo estruturado (SET, ARRAY, RECORD ou FILE). "Compactando" as estruturas de dados, obtemos vantagens consideráveis no uso da memória, especialmente em computadores com palavras longas.

O tipo string

A única ocasião em que você realmente precisa se preocupar com a compactação é ao usar constantes de strings literais. Um string de caracteres entre aspas é indexado a partir de 1 (e não de 0); portanto, declara-se implicitamente como:

```
PACKED ARRAY [1 .. N] OF CHAR
```

onde N é o número de caracteres no string. Considere este programa:

```
PROGRAM PACK STRING;
CONST
  PASCAL = 'PASCAL';
TYPE
  STRING = PACKED ARRAY [1 .. 10] OF CHAR;
VAR
  S     : STRING;
BEGIN
  S := PASCAL          ;
  S := 'EXCESSIVOS CARACTERES';
  S := 'PASCAL         '
END.
```

No núcleo do programa, as duas primeiras instruções são ilegais devido à incompatibilidade dos tamanhos, mas a terceira é admissível (quatro espaços usados como caracteres de preenchimento). Se a definição do tipo STRING não estivesse compactada, todas as atribuições para literais seriam incompatíveis. Na prática, contu-

do, essas restrições não constituem grande problema. Embora o PASCAL admita oficialmente o uso dos procedimentos READ e WRITE somente para entrada e saída de estruturas completas de e para arquivos externos, é possível exibir diretamente no vídeo tipos STRING literais ou de variáveis. Tente escrever um programa usando uma declaração STRING semelhante ao último exemplo e incorpore a instrução

```
REPEAT
  WRITE ('STRING (T P/ TERMINAR) : ');
  READLN (S);
  WRITELN ('O STRING E : ''S,'')
UNTIL S [1] IN ['T','t']
```

Depois de iniciar a execução desse programa, teste-o para verificar os seguintes pontos:

- Os espaços significativos ou as tabulações estão sendo ignorados?
- O que acontece quando a linha é maior que o tamanho do string?
- O que conterá S se apenas um caractere for introduzido?
- E se apenas [Return] for teclado?
- Você conseguiria incluir um código adequado para preencher com espaços elementos não significativos?

Restrições de memória

No PASCAL, não há qualquer limitação quanto ao tamanho ou número de dimensões que você poderá especificar para uma matriz. No entanto, seu computador pode ser desculpado por não aceitar definições como

```
TIPO GRANDE = ARRAY [INTEGER] OF SET OF
  CHAR;
  {SOLICITA NO MINIMO 64K x 128 BYTES!}
AINDAMAIOR = ARRAY [1 .. 1000] OF
  RECORD
    NOME,
    SOBRENOME : STRING;
    ENDERECO : ARRAY [1 .. 5] OF STRING;
    PRESENCA : SET OF PALESTRAS;
    {ETC.}
  END; {AINDAMAIOR}
```

O tamanho físico da memória limitará uma programação tão ambiciosa. Isso mais uma vez reitera a vantagem de se poder especificar a faixa dos tipos escalares, reduzindo ao máximo qualquer improdutividade no armazenamento.

O crivo de Eratóstenes

Este programa — baseado no método para gerar números primos, desenvolvido pelo filósofo e matemático grego Eratóstenes (séc. III a.C.) — tem sua velocidade de execução aumentada quando se ignoram os números pares e emprega-se uma matriz de "flags" (8.192 elementos para os primos até 16.384). Assim, torna-se necessária uma capacidade de memória de pelo menos 8 Kbytes; talvez 32 Kbytes nas linguagens que não possuem operadores booleanos de byte único e integers de 4 bytes. No PASCAL, contudo, todo o conjunto exige apenas 16.384 bits (2 Kbytes), permitindo expressar o algoritmo original de Eratóstenes com mais clareza:

```
COLOQUE TODOS OS NUMEROS (1 .. MAX) NUM CRIVO
TOME O NUMERO 1 (PRIMO POR DEFINICAO)
{IMPRIMA SE DESEJADO}
REPEAT
  PEGUE O MENOR NUMERO DO CRIVO
  {IMPRIMA ESSE PRIMO}
  REMOVA TODOS OS SEUS MULTIPLOS DO CRIVO
UNTIL (O CRIVO ESTEJA VAZIO)
```

Um conjunto assim só pode ser manipulado por computadores de grande porte, mas é possível uma matriz de conjuntos menores (com 100 ou 1.000 elementos, dependendo de seu compilador). Determina-se o índice de matriz de cada conjunto por meio de uma divisão integer, e representa-se sua inclusão no conjunto por N MODULO 100 ou 1.000.

```
PROGRAMA    PRIMOS    (OUTPUT);
{ENCONTRA NUMEROS PRIMOS PELO ALGORITMO
  DE ERATOSTENES}
CONST
  TAMSET      =     100;    {IMPLEMENTACAO}
  PREDTAMSET  =      99;    {TAMSET - 1}
  MAIORPRIMO  =   16383;    {PARA MAXIMTS = 32767}
  LISTAMAX    =     163;    {MAIORPRIMO DIV TAMSET}

TYPE
  FAIXAPRIM    =   1 .. MAIORPRIMO;
  DIMENSAO     =   0 .. LISTAMAX;
  FAIXASET     =   0 .. PREDTAMSET;
  CRIVO        =   SET OF FAIXASET;
  ERATOSTENES  =   ARRAY [DIMENSAO] OF CRIVO;
  CARDINAL     =   0 .. MAXINT;

VAR
  CRIVOS       :   ERATOSTENES;
  CONTADOR,
    MULTIPLO   :   CARDINAL;
  INDICE       :   DIMENSAO;
  N            :   FAIXAPRIM;
  NUMERO       :   0 .. PREDTAMSET;

BEGIN
  WRITELN;
  WRITELN ('CRIVO DE ERATOSTENES' : 50);
  WRITELN ('= = = = = = = = = = = = = = =' : 50);
  WRITELN;
  WRITELN;

  FOR INDICE := 0 TO LISTAMAX DO
    CRIVOS [INDICE] := [ 0 .. PREDTAMSET];
      {COLOCA TODOS OS NUMEROS NO CRIVO}

  CRIVOS [0] := [2 .. PREDTAMSET];
  {WRITELN (1);} {PRIMO POR DEFINICAO}
  CONTADOR := 1;
  FOR N := 2 TO MAIORPRIMO DO
    BEGIN
      INDICE := N DIV TAMSET;
      NUMERO := N MOD TAMSET;

      IF NUMERO IN CRIVOS [INDICE] THEN
        BEGIN
          CONTADOR := SUCC (CONTADOR);
          {WRITELN (N);}
          CRIVOS [INDICE] := CRIVOS
            [INDICE] - [NUMERO];
          MULTIPLO := N + N;

          WHILE MULTIPLO <= MAIORPRIMO DO
            BEGIN
              INDICE := MULTIPLO DIV TAMSET;
              CRIVOS [INDICE] := CRIVOS [INDICE]
                - [MULTIPLO MOD TAMSET];
              MULTIPLO := MULTIPLO + N
            END
        END
    END;

  WRITELN;
  WRITELN (CONTADOR : 25,'PRIMOS ENCONTRADOS.')

END.
```

APLICATIVOS

Advogados, médicos, proprietários de restaurantes e outros profissionais necessitam de programas específicos — os aplicativos —, que são adaptados de pacotes de software genérico ou, então, desenvolvidos especialmente.

Perfis do mercado
Os aplicativos respondem às necessidades de grupos específicos de usuários e são vendidos a grandes setores do mercado. No diagrama, essas áreas de aplicação relacionam-se aos interesses de uma parcela significativa dos usuários. O eixo vertical representa o nível de complexidade da aplicação — os mais baixos são os métodos manuais, a seguir vem o software pronto e, por fim, o software aplicativo. Os perfis de uso mostram que o software pronto se presta muito bem aos requisitos comerciais, mas também indicam que usuários especializados precisam adaptar os pacotes de software já existentes ou criar os seus próprios.

Apesar da quantidade de programas disponíveis no mercado, a maioria apresenta um grande inconveniente: é abrangente demais. Assim, existe pouca probabilidade de que sejam imediatamente aplicáveis a um problema específico. Duas alternativas apresentam-se ao usuário: adaptar o problema ao software existente — o que pode não ser satisfatório —, ou modificar o programa de modo que atenda a suas necessidades.

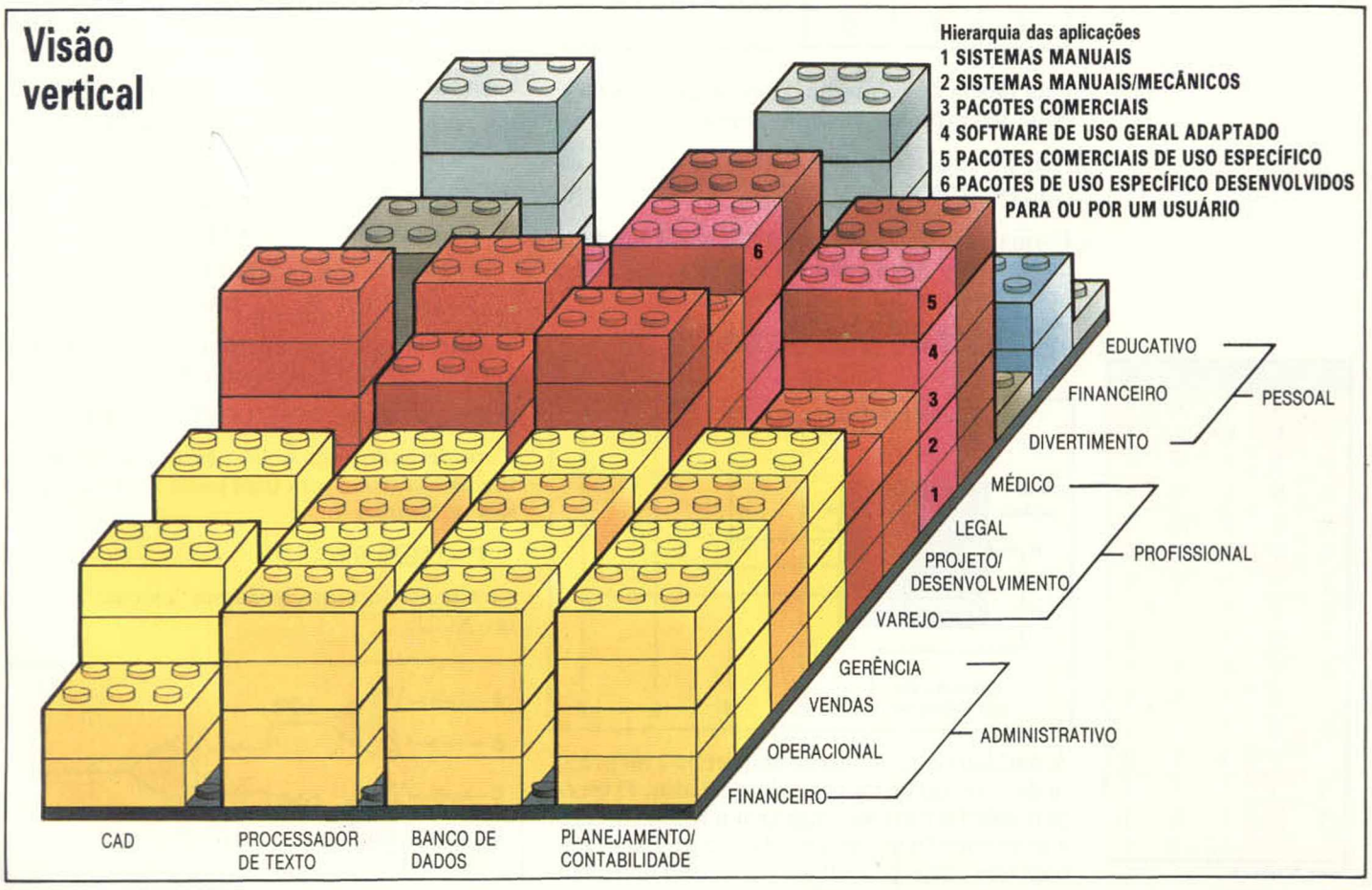
Uma terceira possibilidade, cada vez mais difundida na Europa e nos Estados Unidos, são as aplicações verticais ou softwares aplicativos verticais, isto é, a criação, por entusiastas da informática ou por firmas especializadas, de programas para solução dos problemas específicos de certos profissionais.

Em São Paulo, especialistas em desenhos por computador desenvolveram um programa que ajuda dentistas a planejar tratamentos ortodônticos. A partir da descrição da arcada dentária do paciente, o programa desenha a involução provável dos defeitos sob condições de uso de aparelhos corretores.

Outro exemplo de software vertical é o aplicativo desenvolvido pelo arquiteto José Eduardo Maluf de Carvalho. Para realizar o projeto de um edifício, o arquiteto ou o engenheiro precisa saber que características a obra deverá ter em obediência às leis de zoneamento.

Cada uma das dezoito zonas em que se divide a cidade de São Paulo possui uma legislação precisa quanto ao tipo de construção permitida, áreas máxima e mínima de construção, recuos etc. Em geral são necessárias três horas para se realizar esse levantamento; mas, com auxílio do programa criado por José Eduardo, abrevia-se esse tempo para apenas cinco minutos.

Essas são algumas respostas criativas à inevitável questão colocada por pessoas que acabaram de adquirir um micro: "Para que ele serve?" Estima-se que, em média, os usuários de micros não utilizam mais do que 10% da capacidade do equipamento. Afinal de contas, investir dinheiro numa máquina e limitá-la a uma única aplicação — seja jogos ou administração de contas pessoais — não é, certamente, tão razoável e divertido quanto explorar ao máximo sua versatilidade.

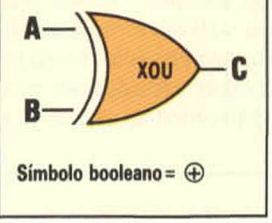


NO NÍVEL

Este curso, que já permite projetar circuitos complexos, mostra agora o desenvolvimento de todo o processo de desenho para um circuito gerador de bits de paridade e um decodificador de prioridades.

Antes de examinar o projeto desses dois aplicativos de nível adiantado, convém analisar em detalhe outra porta lógica importante — a porta OU exclusivo (XOU). Embora essa porta já seja conhecida, ainda não foram investigados nem a álgebra booleana, nem os símbolos do diagrama de circuito para ela:

Tabela de validação			Símbolo do circuito
A	B	C	
0	0	0	
0	1	1	
1	0	1	
1	1	0	



Pela tabela de validação, nota-se que a saída C pode ser expressa de duas formas:

a) $C = A \oplus B = \overline{A}.B + A.\overline{B}$

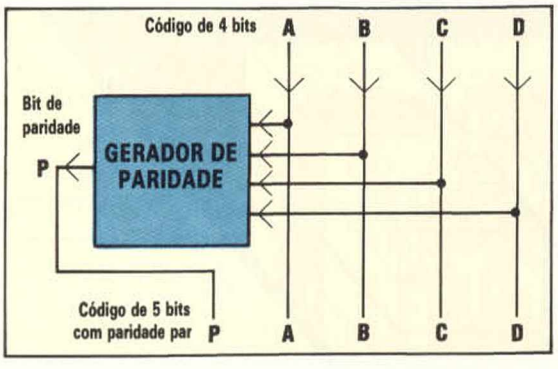
b) $\overline{C} = \overline{A \oplus B} = \overline{A}.\overline{B} + A.B$

Forma-se a segunda expressão considerando-se os casos em que C não é 1 (ou seja, é 0). Essa porta terá uso no primeiro aplicativo.

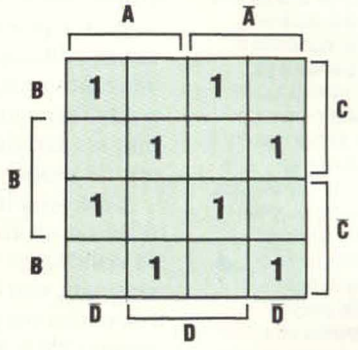
Gerador de bits de paridade

A	B	C	D	P
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

Tabela de validação



A paridade é um conceito importante no projeto de sistemas de transmissão de dados. O bit de paridade (ver o diagrama) de um código binário é acrescentado ao restante do código para fazer com que todos os códigos transmitidos tenham um número par (paridade par) ou ímpar (paridade ímpar) de dígitos 1. O bit de paridade funciona como um sistema de verificação para nos certificarmos de que houve transmissão correta. O circuito desenhado admite um código de 4 bits e apresentará o bit de paridade apropriado. Com uma pequena modificação, o circuito também poderá funcionar como verificador de paridade de dados recebidos. A tabela de validação para esse circuito encontra-se na margem. A representação desses valores em um mapa-k é



O padrão simétrico apresentado pelo mapa-k não permite simplificação porque não é possível formar grupos. A expressão resultante para P será

$$P = \overline{A}.\overline{B}.\overline{C}.D + \overline{A}.\overline{B}.C.\overline{D} + \overline{A}.B.\overline{C}.\overline{D} + \overline{A}.B.C.D + A.\overline{B}.\overline{C}.\overline{D} + A.\overline{B}.C.D + A.B.\overline{C}.D + A.B.\overline{C}.\overline{D}$$

Agrupando os termos em vermelho e em azul, simplificamos a expressão para

$$P = (\overline{A}.\overline{B} + A.B).(\overline{C}.D + C.\overline{D}) + (A.\overline{B} + \overline{A}.B).(\overline{C}.\overline{D} + C.D)$$

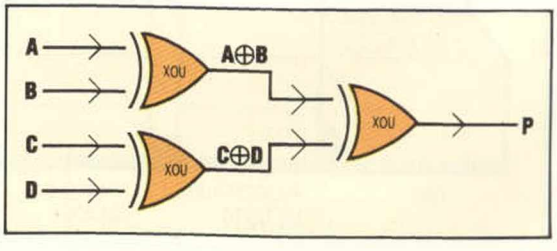
Substituindo as expressões para a porta XOU apresentadas no início, obtemos

$$P = \overline{(A \oplus B)}.(C \oplus D) + (A \oplus B).\overline{(C \oplus D)}$$

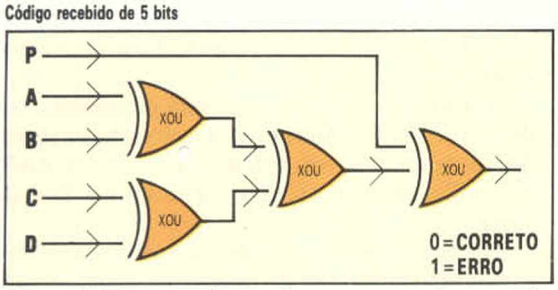
Considerando cada termo entre parênteses como uma entrada para uma porta XOU, a expressão se reduz a

$$P = (A \oplus B) \oplus (C \oplus D)$$

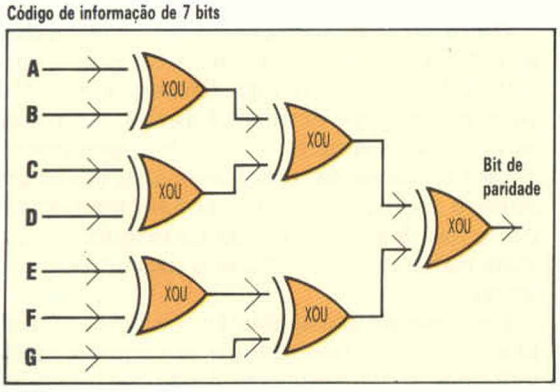
e o circuito formado será uma "cascata" de portas XOU:



Esse circuito funciona como verificador de paridade quando se acrescenta outra porta XOU para comparar os dois bits: o de paridade recebida é o gerado na extremidade receptora.



Na prática, a maioria dos computadores utiliza, para a transmissão de dados, o código ASCII, de 8 bits (7 de informação e 1 de paridade par). É fácil imaginar um gerador de bits de paridade para códigos ASCII:



Decodificador de prioridade

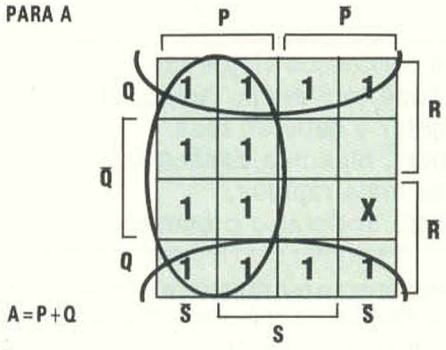
Muitos computadores utilizam sistemas de "interrupção com prioridade" para controlar o fluxo de dados de e para os periféricos. Nesses sistemas, o periférico emite um sinal que interrompe a atividade da CPU. Quando ocorrem sinais simultâneos, segue-se uma ordem de prioridade, a fim de que a CPU "atenda" o periférico mais importante em primeiro lugar. O decodificador de prioridade que projetaremos unirá quatro periféricos num circuito capaz de identificar seus sinais e acionar um sistema de prioridade quando emitidos ao mesmo tempo.

No caso de quatro periféricos, são necessárias duas linhas de saída e uma terceira para requisitar a interrupção. Considere os quatro periféricos P, Q, R e S, tendo P a prioridade máxima e S a mínima. A e B são linhas de saída para identificar o periférico. Z requisita a interrupção. A tabela de validação para o decodificador pode ser construída com o uso de um X para condições que "não interessam" ao decodificador:

P	Q	R	S	A	B	Z
0	0	0	0	X	X	0
0	0	0	1	0	0	1
0	0	1	X	0	1	1
0	1	X	X	1	0	1
1	X	X	X	1	1	1

Para compreender essa tabela, examine a última linha. No caso, P indica uma interrupção e, como se trata do periférico com prioridade máxima, não importa se os outros também estão sinalizando.

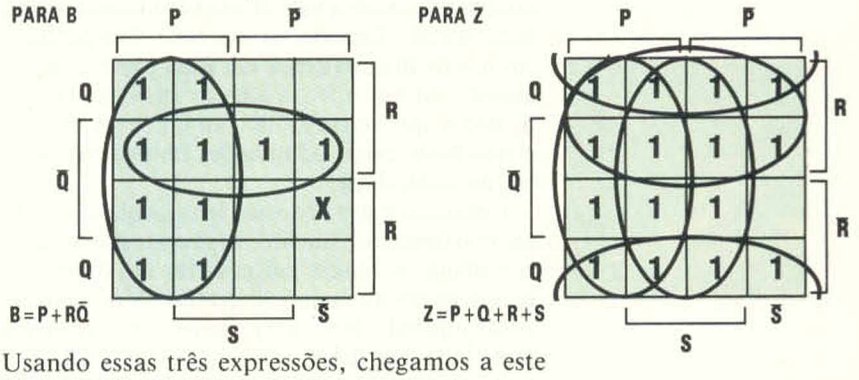
As três linhas de saída do circuito devem ser analisadas independentemente. Começando por A, o mapa-k será



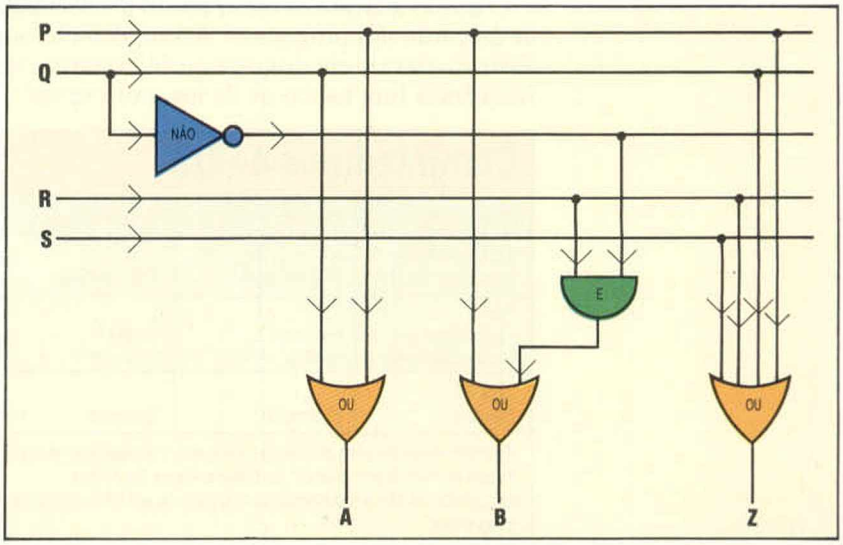
O caso que "não interessa" do lado da saída A está representado por X no mapa-k, mas os que "não interessam" do lado da entrada são manipulados de modo diferente. Vejamos a situação em que P é 1 e Q, R e S são condições que "não interessam". Devemos preencher todos os quadros no mapa-k em que P é 1 — já há oito casos ao todo. Pelo mapa-k, obtém-se a expressão simplificada

$$A = P + Q$$

De modo semelhante, para B e Z, os mapas-k são



Usando essas três expressões, chegamos a este desenho de circuito:



MÉTODOS DE OTIMIZAÇÃO

Com os processos vistos até agora, você terá maior facilidade em projetar e depurar seus programas, mas não conseguirá torná-los mais rápidos. Veja como acelerar o processo.

Uma programação estruturada e o recurso a fluxogramas facilitam a elaboração e o uso de programas, mas não aumentam sua eficiência. Muitas vezes, para que estes sejam executados com mais rapidez e menos memória, é necessário sacrificar a clareza dos métodos utilizados. Então, sempre que otimizamos um bloco de programa de modo a torná-lo mais rápido, ele se torna mais difícil de ler, entender e corrigir.

A lenta execução inerente a linguagens interpretadas, como o BASIC, acarreta situações em que os programas rodam num ritmo inaceitável. E o modo mais eficiente de acelerar um programa em BASIC é compilá-lo. No entanto, poucos micros admitem o uso de um verdadeiro compilador BASIC. Existem no mercado compiladores em discos magnéticos e em fitas cassete, mas a maioria admite apenas o BASIC INTEGER (versão do BASIC que só trabalha com números inteiros) e isso pode exigir adaptações no programa antes da compilação.

Compilar é um processo lento, sobretudo durante o desenvolvimento do programa e quando o sistema se baseia em cassete. O compilador ocupa parte da memória destinada ao usuário; então, quanto mais abrangentes seus recursos, mais bytes da RAM ele ocupará na área destinada ao programa a ser compilado.

O acesso a arquivos reduz muito a velocidade de execução dos programas. Assim, entre os que lêem ou gravam em disquete ou fita cassete com freqüência (um banco de dados, por exemplo), a lentidão é inevitável. Num arquivo de acesso aleatório de um disquete, a recuperação de um registro demora, em média, 1/4 de segundo. Em arquivos seqüênciais, demora ainda mais (conforme o comprimento do arquivo). Com fitas cassete, o problema se agrava, pois os acessos ficam ainda mais lentos.

Quando essas demoras causam problemas, podemos reduzir o número de acessos lendo vários dados de uma só vez, armazenando-os na RAM e atualizando os arquivos apenas no fim da sessão.

Em geral, os programas interativos apresentam o inconveniente de deixar o usuário olhando para uma tela inerte por vários segundos. Uma solução parcial consiste em reorganizar o programa de modo que os arquivos sejam lidos e gravados enquanto o usuário está ocupado com outra coisa (lendo instruções no vídeo, por exemplo).

Outra causa da lentidão é a aritmética usada quando se trabalha com números reais — aritmética de ponto flutuante. (Números reais são aqueles com casas decimais à direita da vírgula; os inteiros — integers — não têm parte fracionária.) Por causa da parte fracionária, buscar um número real na memória e executar uma operação aritmética com ele exige da máquina muito mais trabalho do que fazer o mesmo com um integer.

Em programas com muitas operações aritméticas, convém substituir todas as variáveis envolvidas por variáveis integer, desde que o micro e a versão do BASIC utilizados permitam essa alteração. Dessa forma, consegue-se uma economia de até 20% no tempo de execução de programas sem muitas operações numéricas. Se o programa recebe, como dados de entrada, números previamente processados, consegue-se reduzir o tempo em até 50%.

Projetar um algoritmo mais rápido está entre as melhores maneiras de acelerar um programa. Procure vários algoritmos antes de se decidir por um. Muitos são publicados em livros e em revistas especializadas em computação.

Inventar algoritmos é questão de criatividade e inspiração. A maioria das versões do BASIC dispõe de várias funções eficientes e rápidas, como: INSTR (abreviatura de in string), que procura um substring dentro de uma variável instring; SGN (abreviatura de signal), que obtém o sinal de um número (−1, se negativo; 0, se nulo; e +1, se positivo); e LOG (abreviatura de logaritmo), que obtém o logaritmo decimal de um número. Às vezes vale a pena checar novamente o manual da máquina e ver quais as funções disponíveis, antes de se dar ao trabalho de reescrevê-las em sua própria versão.

Funções definidas pelo usuário, implementadas com o comando DEF FN (disponível em quase todas as versões do BASIC), também são executadas com rapidez. Esse comando é mais útil em programas com uma seqüência repetida de manipulação de strings, onde ele pode subs-

Compiladores BASIC

Para micros	Fabricante	Gravado em
TK 85 e CP 200 Mcoder	Intercomputer	Fita cassete
Apple Task Compiler Hidden Compiler	 Microsoft Hidden	 Disquete Disquete
TRS-80 Bascom	 Microsoft	 Disquete

Existem várias versões do Mcoder. Uma das mais aperfeiçoadas é citada no livro *Super BASIC TK*, da Editora Aleph. Os outros compiladores são encontrados no mercado ou em bibliotecas de programas.

tituir uma chamada de sub-rotina. As rotinas se tornam mais rápidas quando escritas em linguagem de máquina. Isso porque as linguagens interpretadas traduzem cada uma das linhas de um programa para o ASSEMBLY à medida que vão sendo encontradas durante a execução do programa — processo bem pouco eficiente.

Codificar instruções em linguagem de máquina evita esse processo de tradução. Mas escrever um programa em linguagem ASSEMBLY é muito mais difícil do que em BASIC, e o custo (em tempo e esforço) da programação nem sempre compensa a economia no tempo de execução. Por outro lado, alguns programas — os que usam gráficos animados, por exemplo — não funcionariam como planejado se fossem escritos somente em BASIC.

Há muitas outras maneiras de conseguir economias menores na velocidade de processamento. Em alguns micros, vale a pena usar uma variável no lugar de um número (por exemplo, MAX em vez de 267,5) para obter um acesso mais rápido aos valores, sobretudo em loops. Empregue letras diferentes para iniciar os nomes das variáveis; se necessário, utilize todo o alfabeto. Procure usar linhas com várias instruções, quando possível. Se o interpretador permitir, elimine dos loops FOR-NEXT a variável contadora da instrução final, isto é, use apenas NEXT em vez de NEXT e o nome da variável contadora. Dentro de um loop, evite calcular o mesmo valor em cada passagem. Em vez disso, calcule o valor fora do loop e incorpore-o como uma variável.

Economizando espaço

A aritmética usada para números integer, além de tempo, economiza espaço. Empregam-se 4 ou 5 bytes para armazenar um número real e apenas 2 para um integer. Isso representa bastante economia, sobretudo quando são usadas grandes matrizes.

Alguns aumentos na velocidade de um programa também economizam espaço. É o que ocorre quando se usam funções residentes ou definidas pelo usuário, se escreve em linguagem ASSEMBLY ou se recorre a linhas com várias instruções. Por outro lado, a compilação tende a aumentar o tamanho de programas pequenos, só economizando espaços em programas grandes.

A eliminação das linhas REM economiza um bom espaço, e o uso de strings de texto mais curtos para mensagens também ajuda. Colocar grandes blocos de texto em arquivos armazenados fora do programa exclui esses blocos quando não necessários (as instruções e os textos de auxílio ao usuário são o que mais ocupa espaço). Remova os espaços dispensáveis de cada linha, use números de linhas pequenos e nomes mais curtos para as variáveis. Se uma matriz precisa ser dimensionada, mas você desconhece seu tamanho exato, não arrisque um número qualquer. Em vez disso, espere até obter a informação necessária e, então, dimensione-a com uma variável, como neste exemplo:

```
10 INPUT "QUANTOS CASOS HA NESSA
   CATEGORIA?";CASOS%
DIM MATRIZ%(CASOS%)
```

Isso se chama "dimensionamento dinâmico" e é um recurso oferecido pelo BASIC, mas não por muitas outras linguagens. Utilize-o ao máximo.

Outra técnica usada em alguns micros é o aumento da área na RAM para o programa. Pode-se fazer isso por meio de comandos como o HIMEM (do Apple). Em geral esses comandos mudam a área da RAM disponível para programas em BASIC e variáveis. Tal mudança serve para armazenar programas em código de máquina num lugar seguro, onde nada será gravado sobre eles. Mas o mesmo comando permite acessar espaço extra da área reservada à memória de vídeo, constituindo uma boa maneira de se conseguir uns quilobytes a mais de RAM quando o que está aparecendo na tela não é importante. Caso não seja possível alterar a área reservada para programas, use a memória de vídeo diretamente, por meio dos comandos PEEK e POKE, nas localizações da memória reservada para ela.

Quando tudo falha e o programa não cabe no espaço disponível, algumas versões do BASIC dispõem de um comando (CHAIN) que permite a um programa passar o controle para outro. Outras versões empregam o comando COMMON, que transfere determinadas variáveis e seus valores atuais para o próximo programa. Quando existente nos micros, o CHAIN — em geral um comando muito simples — possibilita a transferência de todas as variáveis (ou de nenhuma) do primeiro programa para o segundo.

Em programas elaborados de maneira estruturada, cada rotina deve ser escrita e testada independentemente. Pode-se também cronometrar a execução de cada uma. Um programa medidor de tempo que roda no TK 90X é o seguinte:

```
100  REM  Use esta secao para estabelecer
110  REM  variáveis que serao utilizadas pela
120  REM  rotina a ser testada (nao
130  REM  se esqueça de dimensionar
140  REM  matrizes e preenche-las com dados
150  REM  realistas). Uma variavel do sistema
          chamada TVCOUNT e usada para
             medir o tempo.
160  POKE  23672,0
170  POKE  23673,0
180  POKE  23674,0
190  LET  inicio = PEEK 23672 + 256*PEEK
          23673 + 65536*PEEK 23674
200  GOSUB  2000: REM a rotina a ser
             cronometrada deve ser
             chamada aqui.
210  LET  fim = PEEK 23672 + 256*PEEK
          23673 + 65536*PEEK 23674
220  PRINT  "A execucao demorou "; (fim-inicio)
          /60," segundos. "
2000  FOR  a = 1 TO 10000: NEXT a
2010  RETURN
```

Com essa rotina é possível experimentar diferentes algoritmos e maneiras de aumentar a velocidade.

LÓGICA EMPRESARIAL

Seguindo a tendência do mercado internacional, em 1985 a Prológica lançou seu micro de 16 bits, o SP 16, compatível com o IBM PC.

Com a diversificação dos produtos, a empresa vem ampliando cada vez mais suas instalações. Desta linha de montagem saem três modelos de impressoras matriciais: P-500, P-720 e P-740.

Fruto da ousadia de dois amigos entusiastas da eletrônica, a Prológica tornou-se, em nove anos, uma das maiores e mais bem-sucedidas empresas do setor de micros pessoais e profissionais.

Uma modesta firma de distribuição de produtos eletrônicos, na tradicional rua Aurora, no centro velho de São Paulo, foi palco, em março de 1976, de importante decisão. Dois amigos, unidos pelo prazer de experimentar ludicamente equipamentos eletrônicos, resolveram comercializar o protótipo da máquina contábil que acabavam de montar. Seis meses depois, Leonardo Bellonzi e Joseph Blumenfeld lançavam sua engenhoca no mercado, passando de distribuidores a produtores de equipamentos eletrônicos.

A tentativa revelou-se um sucesso: em dois anos venderam-se mais de 5.000 unidades, confirmando a possibilidade comercial da nova associação. Assim nasceu a empresa que, a partir de 1980, ocuparia destacada posição entre as produtoras nacionais da área de informática — a Prológica Indústria e Comércio de Microcomputadores Ltda.

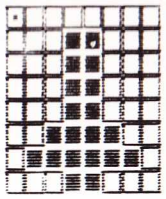
Nove anos depois, em 1985, a Prológica ocupava o terceiro lugar na classificação das empresas nacionais do setor, empregando mais de 1.500 funcionários. Esse desempenho invejável deve-se em grande parte à filosofia, adotada desde sua fundação, de vender não apenas equipamentos mas, sobretudo, soluções. Todos os produtos foram desenvolvidos a partir de pesquisas sobre as necessidades de seus clientes potenciais.

A Prológica conta hoje com extensa rede de pontos de venda e com programas de treinamento oferecidos a clientes e usuários.

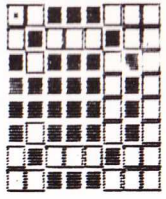
Depois do primeiro sistema contábil, lançado no final da década de 70, a empresa concentrou suas atividades na produção de microcomputadores pessoais (40%), sistemas profissionais (40%) e periféricos (20%). A linha de micros inclui desde modelos mais simples, como o CP 200 e o CP 300, até equipamentos com recursos avançados, como o CP 400 Color e o carro-chefe da linha, o CP 500. No setor dos equipamentos profissionais, a Prológica produz o Sistema 700 e o Super 700, ambos de 8 bits, e o SP 16, um micro modular de 16 bits compatível com o IBM PC. Impressoras matriciais, unidades de disco rígido tipo Winchester e unidades de disco flexível completam sua linha de produtos.

Acompanhar as modificações de um mercado dinâmico como o da informática implica investimento constante na criação de novos produtos. Por isso, trezentos engenheiros e técnicos da Prológica dedicam-se exclusivamente à pesquisa tecnológica, que consome cerca de 10% do faturamento da empresa.

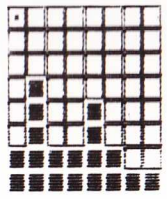
Utilize as seguintes figuras para redefinição de caracteres:



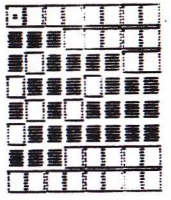
Míssil



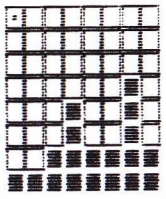
Proa da nave Polaris



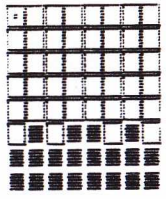
Lado direito da base



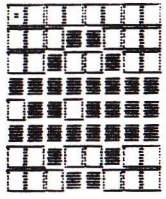
Cauda da nave Polaris



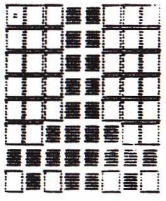
Lado esquerdo da base lançadora de mísseis



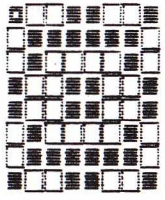
Fundo do oceano



Corpo da nave Polaris

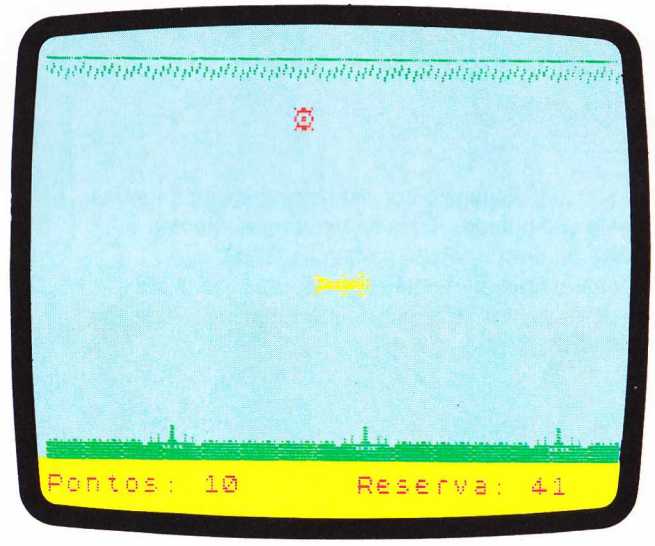


Lançador de mísseis

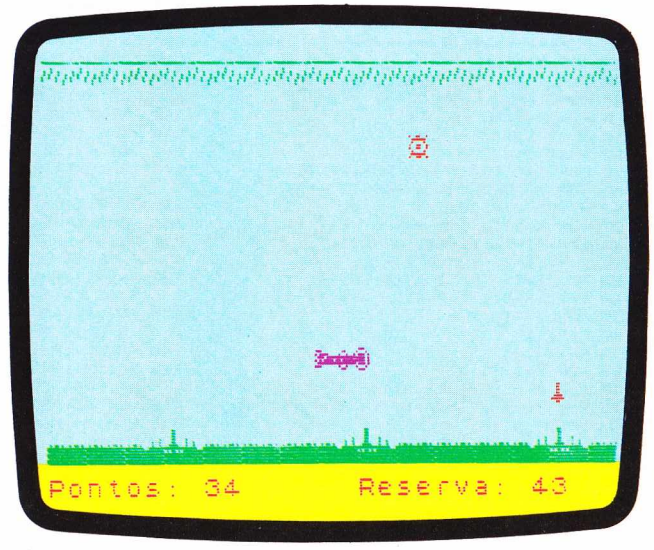


Nave Chenggers

```
,i;" ": IF o THEN GOTO 210
 120 IF RND<.01 AND u=0 THEN LET
 x=INT (RND*30): LET y=INT (RND*
9+1): LET u=1
 130 IF INKEY$="" THEN GOTO 320
 140 IF INKEY$="0" THEN LET j=25
 150 IF INKEY$="1" THEN LET j=6
 160 IF INKEY$="6" THEN LET j=15
 170 LET o=1: LET r=r-1
 180 SOUND .01,30: LET b=17
 190 PRINT AT 20,30;" ";AT 20,25
;r;" "
 210 LET b=b-1
 220 IF b<>a THEN GOTO 240
 230 IF ATTR (b,j)<>7 THEN GOTO
360
 240 PRINT AT b,j;"";AT b+1,j;"
"
 250 IF b>1 THEN GOTO 320
 260 LET o=0
 270 PRINT AT b,j;" "
 280 IF NOT r THEN GOTO 410
 290 GOTO 320
 320 IF u THEN PRINT AT y,x;" ";
 LET x=x+(x<31)-(x AND x>30): LE
T y=y-(RND#2 AND y>3)+(1 AND RND
<.1): SOUND .0009,55: PRINT AT y
,x;""
 330 NEXT i
 340 PRINT AT a,29;"
```

Campeão de tiro
Sua reserva de munição é grande: cinqüenta mísseis, para você destruir o maior número de naves possível e estabelecer um recorde de pontos.




```
 350 GOTO 100
 360 PRINT AT b+1,j;" "
 370 IF ATTR (b,j)=6 THEN LET o=
0: PRINT AT a,l+1;"" FOR f=0
 TO 10: SOUND .09,-f: NEXT f: PR
INT AT a,l+1;" ": LET d=d+(16-
a)#(4-((j-6)/10+1))
 380 IF ATTR (b,j)=2 THEN LET d=
d+(31-x)#3+y#10: PRINT AT y,x;""
": FOR f=0 TO 10: SOUND .09,30+f
: NEXT f: PRINT AT x,y;" ": LET
u=0
 390 PRINT AT 20,8;d
 400 IF r THEN GOTO 100
 410 IF d<=c THEN GOTO 430
 420 PRINT AT 9,5;"Estabelecido
novo Record": LET c=d: GOTO 440
 430 PRINT AT 9,8;"O Record perm
anece"
 440 PRINT AT 11,16-(10+LEN STR$
 c)/2;"de ";c;" Pontos .": PRINT
 AT 13,7;"Quer jogar de novo ?":
 IF INKEY$<>"s" THEN SOUND RND,R
ND#60: GOTO 440
 450 GOTO 90
 460 PRINT AT 19,0;"
";AT 18,0;"
";AT 20,0;"Pontos: 0";TAB 16;"Res
erva: 50": RETURN
```

PAPAGAIO

PARA LINHA APPLE

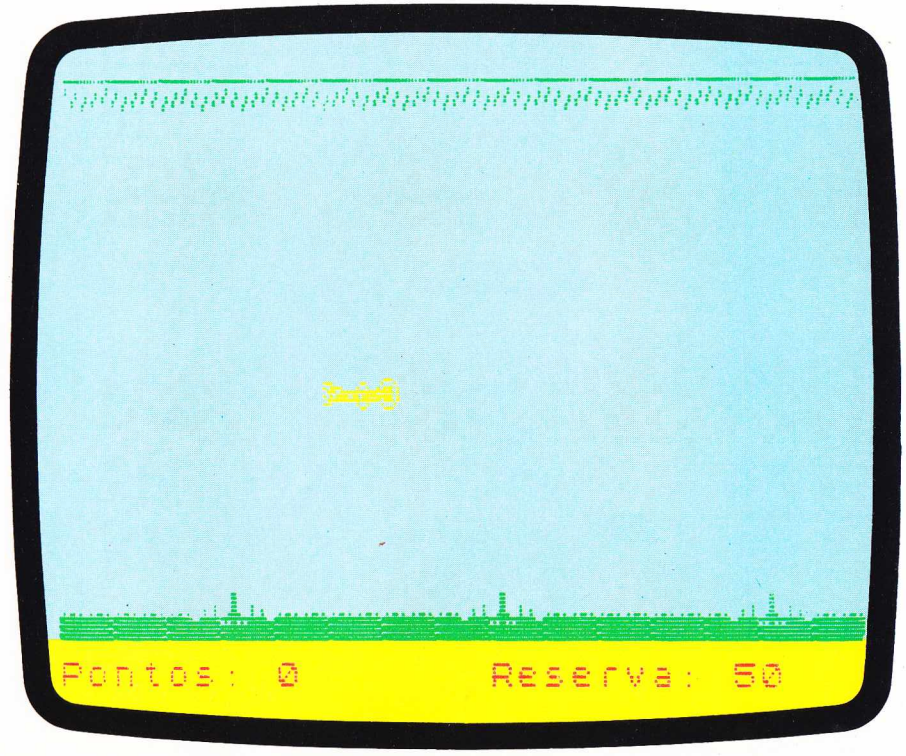
Jogo de competição muito simples, Papagaio consiste na repetição de letras sugeridas aleatoriamente pelo computador. Trata-se de um passatempo ideal para crianças, que, além de divertir, ajuda na familiarização com o teclado. Este programa permite também testar reflexos, uma vez que a repetição deve ocorrer o mais rapidamente possível. Ganha quem consegue o maior número de letras no menor tempo.

```
 10 REM*********************************
 12 REM*       PARA LINHA APPLE        *
 14 REM*                               *
 16 REM*          PAPAGAIO             *
 18 REM*********************************
 19 GOSUB 700
 20 DIM X(11),Y(11),A(11),CO$(9)
 30 HOME :HGR :HCOLOR=3
 40 FOR I=0 TO 10
 50 READ X(I),Y(I),A(I)
 60 NEXT
 70 FOR I=1 TO 9:READ CO$(I):NEXT
100 REM ** DESENHA PAPAGAIO **
110 HPLOT X(0),Y(0)
120 FOR I=1 TO 9
130 X1=X(I-1):X2=X(I):A=A(I)
140 Y1=Y(I-1):Y2=Y(I)
150 FOR X=X1 TO X2 STEP SGN(X2-X1)
160 Y=(((X-X1)/(X2-X1))^A)*(Y2-Y1)+Y1
170 HPLOT  TO X,Y
180 NEXT X
190 NEXT I
200 FOR R=3 TO 4
210 FOR A=0 TO 6.28 STEP .2
220 X=X(10)+R*SIN(A):Y=Y(10)-R*COS(A)
230 HPLOT X,Y
240 NEXT A
250 NEXT R
500 REM ** PROGRAMA PRINCIPAL **
505 HOME
510 VTAB (23):PRINT "      ESTA PRONTO
    ?";
520 GET A$:PRINT :PRINT
530 Q$=CHR$(65+INT(RND(1)*26)):PRINT Q
    $" ";
540 TI=TI+1:IF TI>350 THEN 600
550 IN=PEEK(-16384):IF IN<128 THEN 540
560 POKE 16368,0
570 IF CHR$(IN-128)<>Q$ THEN 540
580 RS=RS+1:GOTO 530
600 PRINT :PRINT "RESULTADO: "RS
610 IF RS>18 THEN RS=18
620 PRINT CO$(INT(RS/2+.5))
630 FOR I=1 TO 1500:NEXT
640 PRINT "DE NOVO ? (S/N) ";
650 POKE -16368,0
660 GET A$:IF A$="" THEN 650
670 IF LEFT$(A$,1)<>"S" THEN TEXT :HOME
    :END
680 RS=0:TI=0:GOTO 500
700 HOME :PRINT TAB(16)"PAPAGAIO"
710 PRINT :PRINT "O OBJETIVO DESTE JOG
    O MEDIR SEUS REFLE-"
720 PRINT :PRINT "XOS, TENTANDO REPETI
    R AS LETRAS SORTE-"
730 PRINT :PRINT "ADAS PELO COMPUTADOR
     O MAIS RAPIDO POS-"
740 PRINT :PRINT "SIVEL."
750 PRINT :PRINT " DIGITE QUALQUER TEC
    LA PARA CONTINUAR";
760 GET A$:RETURN
999 REM ** DADOS PARA GRAFICO **
1000 DATA  95,130,0,90,110,1.5,75,115,.
    15,90,100,1.5,60,125,2.5,80,70,.25
    ,100,50,.4,130,75,4,115,110,.4,110
    ,130,1.5,90,68,0
1030 DATA      "BEM RUIM","AINDA NAO","M
    AL !!!","VOCE ESTA APRENDENDO"
1040 DATA  "QUASE LA'","MUITO BEM","EXC
    ELENTE","FANTASTICO","VOCE E' O MA
    XIMO"
```


S.O.S. ATLANTIS

PARA TK90X

Você é o Guardião Supremo de Atlantis e deve defender essa cidade submarina do ataque de naves inimigas. Para cumprir sua missão use as três bases: com a tecla [1] você lança mísseis da primeira base, com [6] da segunda, e com [0] da terceira. Em cada tecla aparecem no máximo duas naves, uma Polaris e uma Chenggers, que você precisa destruir. O jogo termina quando sua reserva de mísseis se esgota. No final, aparece na tela a contagem dos pontos e um convite para superar sua marca em nova tentativa como defensor da cidade.



Instruções para digitação:

110 A nave Polaris deve estar na cor amarela.
240 O míssil deve ser vermelho.
320 A nave Chenggers é vermelha.
370 A Polaris deve estar na cor magenta e piscando.
380 A Chenggers está piscando.

```
10 REM Para o TK 90X
20 REM S.O.S. Atlantis
40 REM  Com ligeiras mudancas
50 REM  roda no TK85,83 e 82
60 REM
70 REM          Boa sorte !
80 LET c=100
 90 BORDER 0: INVERSE 0: INK 7:
 PAPER 0: CLS : FOR f=0 TO 254 S
TEP 3: PLOT f,170+8*SIN (f/6*PI):
DRAW 1,3: NEXT f: LET x=0: LET y
=0: LET o=0: LET d=0: LET r=50:
LET v=0: LET j=15: GOSUB 460
 100 LET a=INT (RND*13+2): FOR i
=0 TO 28
 110 SOUND .00039,30: PRINT AT a
```